

UNIVERSITÉ DE GENÈVE
Département d'Informatique

FACULTÉ DES SCIENCES
Codirecteurs de thèse: Professeur P. Zanella
Docteur B. Chopard

Lattice Wave Automata

From Radio Waves to Fractures Propagation

THÈSE

présentée à la Faculté des Sciences de Université de Genève
pour obtenir le grade de Docteur ès sciences, mention Informatique

par

Pascal Olivier LÜTHI

de

Rüderswil (BE)

Thèse N° 2984

GENÈVE
1998

La Faculté des sciences, sur le préavis de Messieurs P. ZANELLA, professeur ordinaire et B. CHOPARD, docteur (Département d'informatique) codirecteurs de thèse et J.-F. WAGEN, docteur (Swisscom), autorise l'impression de la présente thèse, sans exprimer d'opinion sur les propositions qui y sont énoncées.

Genève, le 18 mars 1998

Thèse -2984-

A handwritten signature in black ink, appearing to read 'E. Doelker', written in a cursive style.

Le Doyen, Eric DOELKER

Everything happens as if...

Avant-propos

This dissertation presents the main subject of my work under the supervision of Dr. Bastien Chopard in the Group for Parallel and Scientific Computing of the Computer Science Department (CUI) at the University of Geneva. The subject concerns original numerical simulation techniques of the generic problem of wave propagation in heterogeneous media. The most important outcomes were published in scientific journals and regularly presented at conferences during the research period between 1993 and 1997. During this period, the approach was several times reinvented or reformulated as it is expected for this kind of research. One of the actual contribution of the present publication is the global and coherent presentation of all the aspects of the approach from the numerical, mathematical and physical point of view. Moreover, our method is likely to be applied in a context very different from the framework it is issued from, and thus I also briefly presents some promising new perspectives.

I did not dedicate a specific chapter to a state-of-the-art presentation of other wave propagation prediction techniques. Neither do I resume basic notions of cellular automata and parallel computing to be found in numerous text books. I choose to concentrated my efforts on our new results and aspects of our approach without forgetting to give significant references for the reader who may go further ahead. We believe and we hope that this may promote an accurate use of our simulation technique and serve as a basis for future studies.

Besides this work, at least three other less extensive researches were undertaken during the same period by the author, and the resulting publications are referred in the text and given in appendix. Although the subjects were completely different, ranging from road traffic problems to micro-biological questions, the same intensive usage of parallel algorithms and parallel computing resources, of generalized cellular automata-like numerical models and of a fundamental, say a physicist's, approach to complexity are characterizing all the works.

I would like to thank Professor Paolo Zanella for accepting me in his de ces perturbationsde ces perturbationsresearch group and giving the opportunity of doing a PhD at the University of Geneva. This work was mainly supported by

the Fond National Suisse de la Recherche. The variety of the effort and successes of these studies offered me a challenging and rewarding time and this is largely due to the exceptional human and scientific quality of Dr. Bastien Chopard. A special thanks is also addressed to Dr. Jean-Frederic Wagen from the Telecom PTT (now SWISSCOM SA) for his help and contributions in various aspects of the work, to Dr. Jeremy J. Ramsden from the Biozentrum Basel, not to forget my PhD students colleagues Alexandre Masselot, Rodolphe Chatagny and Marc Martin and other members of the CUI for their inestimable help and presence during these years.

Pascal Olivier Luthi

Geneva, December 1997

Résumé

Introduction

Cette thèse présente le sujet de travail principal de l'auteur, effectué entre 1993 et 1997, sous la direction de Dr Bastien Chopard dans le groupe du professeur Paolo Zanella (Calcul Scientifique et Parallèle) au Département d'Informatique de l'Université de Genève. Le but de ce travail est l'établissement d'un modèle numérique précis et efficace pour simuler la propagation d'ondes dans un univers discret à l'aide d'ordinateurs parallèles. Les résultats obtenus au cours de cette recherche ont été présentés lors de plusieurs conférences et ont fait l'objet de plusieurs publications dans des revues scientifiques.

A côté de ce sujet, d'autres problèmes ont été traités pendant la période de recherche. Ces autres sujets abordent le problème de la formation de structures macroscopiques lors de la précipitation dans des systèmes de réaction-diffusion, la simulation microscopique du trafic routier, ou encore l'étude de la dynamique de différenciation cellulaire lors des premiers stades de développement de l'embryon. Bien que fondamentalement différents, ces problèmes ont été traités avec une même approche de la "complexité". La démarche consiste à ne retenir des phénomènes étudiés que les caractéristiques microscopiques les plus essentielles puis à les traduire dans un modèle microscopique discret. Ces modèles discrets sont finalement réalisés sur des ordinateurs parallèles et analysés intensivement.

La compréhension des divers phénomènes physiques, biologiques ou sociaux-économiques par l'intermédiaire de simulations numériques profite largement du formidable progrès technologique offrant jour après jour des ordinateurs plus puissants. Mais si la recherche scientifique en général ne peut aujourd'hui plus se passer de l'outil informatique, l'usage qu'elle en fait varie passablement d'une communauté de chercheurs à l'autre. Notre approche par "automates cellulaires" se distingue des autres démarches par le niveau d'abstraction de ses modèles. Nos algorithmes ne résolvent pas des systèmes d'équations différentielles décrivant le phénomène, mais réalisent concrètement des modèles microscopiques discrets. Ces modèles ne cherchent pas à imiter au mieux la réalité microscopique, défini-

tivement trop complexe, des phénomènes mais représentent une réalité virtuelle c'est-à-dire une microscopie fictive simplifiée bien adaptée à l'outil informatique mais capable d'imiter la réalité lorsqu'on l'observe à une plus grande échelle.

Modélisation d'ondes sur réseau

La propagation d'ondes de quelconque nature est un phénomène universel. Le phénomène peut être observé à toutes les échelles et dans tous les milieux. Bruits se propageant dans l'air, vagues animant la surface des océans ou encore lumière parcourant le vide, sont autant de manifestations différentes, parmi d'autres encore, du même processus ondulatoire.

Si les ondes sont si générales c'est qu'elles doivent refléter quelques propriétés fondamentales du système dynamique dans lequel elles se développent. Ces propriétés sont la causalité et la réversibilité par rapport au temps. Mathématiquement, une onde se décrit par l'équation d'onde différentielle du deuxième ordre suivante:

$$\partial_t^2 I \Leftrightarrow c^2 \nabla^2 I = 0$$

La grandeur I représente une "information" susceptible de se propager à la vitesse c dans un milieu immobile. Malgré la diversité des phénomènes évoqués plus haut, l'équation d'onde associée a toujours la même forme. Elle ne donne donc aucune information sur les détails physiques sous-jacents, ceux-ci sont contenus dans un jeu d'équations du premier ordre plus fondamentales dont on peut déduire facilement l'équation d'onde. Nous choisissons, comme paradigme à notre modèle, le jeu suivant:

$$\begin{aligned} \Pi_{\alpha\beta} &= c^2 I \delta_{\alpha\beta} && \text{Une équation d'état} \\ \partial_t I + \partial_\alpha J_\alpha &= 0 && \text{Conservation de } I \\ \partial_t J_\beta + \partial_\alpha \Pi_{\alpha\beta} &= 0 && \text{Conservation de } J_\beta \end{aligned}$$

La tâche consiste donc maintenant à inventer un monde fictif, microscopique et discret correspondant à ce paradigme continu. Pour des raisons pédagogiques il est plus simple, dans un premier temps, de décrire le modèle microscopique, puis de montrer dans quelle limite il représente bien le phénomène de propagation d'ondes compris dans le paradigme donné plus haut.

Le monde *a priori* discret dans lequel évolue notre "onde" est idéalement représenté par un réseau régulier ou une grille remplissant l'espace. À chaque instant discret de l'évolution, le système est entièrement défini par la donnée d'un ensemble discret de valeurs réelles $\{f_0, f_1, f_2, \dots\}$ défini en chaque point de la grille, ou en chaque point du réseau. L'évolution du système pendant une itération se décompose dans une phase de **collision** et une phase de **propagation**. Durant la phase de collision, en chaque site, chaque f_i se transforme selon

une combinaison linéaire des f_i du site, alors que durant la phase de propagation chaque $f_{i \neq 0}$ se propage vers le site voisin indiqué par la direction du réseau \vec{v}_i ; $v = |v_i|$ est la taille de la maille du réseau. La grandeur f_0 représente une valeur qui ne change pas de site au cours de la phase de propagation. L'évolution du système peut ainsi s'écrire sous la forme compacte suivante, caractéristique des modèles BGK sur réseau:

$$f_i(t + \tau, \vec{r} + \tau \vec{v}_i) \Leftrightarrow f_i(t, \vec{r}) = \frac{1}{\xi} \left(f_i^{(0)}(t, \vec{r}) \Leftrightarrow f_i(t, \vec{r}) \right)$$

où $f_i^{(0)}$ - combinaison linéaire de f_i - et ξ définissent entièrement notre modèle. Pour établir le parallèle entre ce modèle discret et le paradigme continu, il faut encore définir l'équivalent discret des grandeurs I et \vec{J} données plus haut:

$$I_{(\text{discret})} \doteq \frac{1}{N+1} \sum_j f_j$$

$$\vec{J}_{(\text{discret})} \doteq \frac{1}{N} \sum_j \vec{v}_j f_j$$

où N est le nombre de coordination, c'est-à-dire le nombre de liens déterminé par le réseau choisi. Si la distribution d'équilibre $f_i^{(0)}$ est donnée par

$$f_i^{(0)} = a_i I + b_i \frac{\vec{v}_i \vec{J}}{v^2}$$

le but est maintenant d'imposer certaines conditions sur les paramètres du modèle $\{a_0, a_1, \dots, b_0, b_1, \dots, \xi\}$ afin de respecter les propriétés fondamentales requises:

$$\begin{aligned} \text{Conservation de } I_{(\text{discret})} &\Rightarrow a_{i \neq 0} = a \text{ et } a_0 + Na = N + 1 \\ \text{Conservation de } \vec{J}_{(\text{discret})} &\Rightarrow b_{i \neq 0} = \mathcal{N} \\ \text{Réversibilité temporelle} &\Rightarrow \xi = \frac{1}{2} \end{aligned}$$

où \mathcal{N} indique la dimension de l'espace. Ainsi ne reste-t-il plus qu'un seul paramètre a et nous montrons qu'il correspond au paramètre c du paradigme: la vitesse de propagation de l'onde. En effet dans la limite du continu $\tau \rightarrow 0$ et $v \rightarrow 0$ on peut montrer, grâce au formalisme de l'analyse multi-échelle, que notre modèle est bien l'équivalent discret du paradigme continu avec la relation suivante:

$$c^2 = \frac{a}{\mathcal{N}} \frac{N}{N+1}$$

Nous disposons donc d'un modèle microscopique discret original permettant de simuler précisément un milieu dans lequel se propage une onde. Le modèle s'applique à des espaces discrets de dimension quelconque pourvu que le réseau

soit régulier et que les liens empruntés par les f_i soit de longueurs égales. Le modèle respecte les symétries fondamentales sous-jacentes à tout phénomène ondulatoire et permet de fixer une vitesse de propagation de l'onde (c'est-à-dire un indice de réfraction) différent en chaque site. Toujours au niveau microscopique, il peut définir des sites absorbants ou réfléchissants ce qui en fait l'outil idéal pour aborder toute sorte de problèmes compliqués impliquant une propagation ondulatoire dans des milieux fortement inhomogènes.

Algorithmes et performances

La simplicité et la généralité de notre modèle doit se traduire en algorithmes performants permettant des simulations précises et rapides d'une très grande variété de phénomènes. De plus, nous verrons que notre problème se parallélise particulièrement facilement c'est-à-dire que l'on peut tirer ainsi le meilleur profit des nouvelles générations d'ordinateurs parallèles; le principe d'un ordinateur parallèle étant de résoudre de façon concurrente plusieurs tâches en même temps partout où cela est possible.

Numériquement, l'état du système est donné par un ensemble de tableaux `if*` et `of*` correspondants aux f_i du modèle. A chaque position dans le tableau correspond un site de notre espace discret et il y a autant de tableaux qu'il y a de f_i . Le noyau de l'algorithme correspond aux deux phases évoquées lors de la description du modèle: collision et propagation. La collision est un calcul purement "local"; elle consiste à effectuer une combinaison linéaire des f_i en chaque point du tableau. La phase de propagation correspond à un déplacement des données dans la mémoire. A l'aide d'un pseudo-langage utilisant les conventions et notations du parallélisme de données (les indices des tableaux sont omis et les opérations sur les tableaux sont effectuées point par point), on obtient pour une itération et avant toute optimisation (les coefficients correspondent au cas particulier sans f_0):

```

of1 = 0.5 * ( if1 + if2 - if3 + if4 )
of2 = 0.5 * ( if1 + if2 + if3 - if4 )
of3 = 0.5 * (- if1 + if2 + if3 + if4 )
of4 = 0.5 * ( if1 - if2 + if3 + if4 )

if1 = CSIFT( if1, DIM=1, SHIFT=-1)
if2 = CSIFT( if2, DIM=2, SHIFT= 1)
if3 = CSIFT( if3, DIM=1, SHIFT= 1)
if4 = CSIFT( if4, DIM=2, SHIFT=-1)

```

Cette façon d'écrire est très compacte mais toute une série d'améliorations peuvent être apportées à ces lignes de codes. Les améliorations permettent de réduire le nombre d'opérations, le nombre de tableaux (c'est-à-dire la mémoire) et le nombre de données à déplacer. Finalement on obtient un code séquentiel optimal dont le noyau ne comporte (avec encore d'autres opérations omises dans le pseudo-code donné plus haut) plus que 15 opérations en virgule-flottante et ne déplace plus que la moitié des données.

Notre modèle peut être considéré comme caractéristique de toute une gamme de problèmes scientifiques réguliers. C'est pourquoi nous nous en sommes servis comme *benchmark* afin de tester et comparer les performances de différentes machines séquentielles et de différents compilateurs. Mais actuellement, les performances des machines séquentielles courantes sont insuffisantes pour effectuer des simulations permettant un suivi interactif de la propagation sur de grands systèmes. Il faut ainsi près de 1 min. 30 sec. pour propager une onde dans un réseau de taille 600×600 à l'aide d'un processeur RISC 10000 (175 MHz, 1125 Mbytes).

Le gain supplémentaire de performance peut être obtenu grâce à la parallélisation de notre algorithme. Notre problème est de type *data-parallèle*: il comporte un grand nombre de données et la solution se décrit bien par une seule liste des opérations qu'il faut effectuer sur chacune des données. Ainsi le modèle de programmation le plus naturel est celui du parallélisme de données. Nous avons testé et obtenu les meilleures performances avec le CMFortran sur sa machine dédiée massivement parallèle: la CM200 (Connexion Machine). Le HPF (High Performance Fortran) est aujourd'hui le seul standard pour réaliser du calcul *data-parallèle* sur des machines multi-processeurs actuelle d'intérêt général. Cependant, malgré la régularité de notre problème, les compilateurs actuels n'offrent que de très médiocres performances. Ainsi faut-il théoriquement près de 27 processeurs RISC 6000 pour obtenir avec HPF, sur la machine IBM SP2, des performances comparables à l'exécution du code séquentiel sur un processeur. C'est la raison pour laquelle nous avons opté pour un modèle de programmation plus "bas niveau": l'échange de messages.

La programmation par échange de messages fait principalement appel à des routines de communications et de synchronisation explicite entre les processeurs. L'usage d'une bibliothèque de routines standard comme PVM (Parallel Virtual Machine) nous a permis d'écrire un code d'une grande portabilité et de mesurer les performances de notre algorithme sur une grande variété de machines allant du cluster de PC "fabrication maison" jusqu'au super-ordinateur CRAY T3D. Le principe de l'implémentation est simple: le domaine de calcul (les tableaux) est découpé en bandes successives et chaque processeur s'occupe d'une bande. Pour calculer les bords de son domaine, chaque processeur doit connaître des données voisines résidant sur le processeur adjacent et c'est précisément là qu'interviennent les routines d'échange de messages. La clé d'une programmation efficace devient ici l'agencement subtile des communications et des synchronisa-

tions afin que temps de calcul et temps de communication se chevauchent le plus possible.

Pour pouvoir comparer des machines si différentes nous avons utilisé un modèle de performance à deux paramètres indépendants de la taille du problème et du nombre de processeurs impliqués. On écrit pour cela le temps T_p d'exécution d'une itération (pour un problème de taille $n \times n$) sur p processeurs de la façon suivante:

$$T_p = a \frac{n^2}{p} + bn$$

On remarque les deux contributions fondamentales: la portion de **calcul** proportionnelle à la taille du domaine n^2 distribuée sur les p processeurs impliqués et la fraction de **communication** proportionnelle à la taille du message que doivent s'échanger les processeurs (n est la longueur d'un bord du domaine). Les paramètres a et b sont ainsi bien représentatifs de la puissance de calcul et l'efficacité de la communication de la machine analysée. Nous avons établi que ce modèle de performance correspond parfaitement à nos mesures pour une grande gamme de machine à mémoire distribuée. Une carte de performance permet de classer de façon originale les machines en fonction de leurs coordonnées (a, b) . Cette carte (voir figure ci-après) contient des zones surprenantes où des machines valant plusieurs millions de dollars en côtoient d'autres plus de 100 fois moins chères.

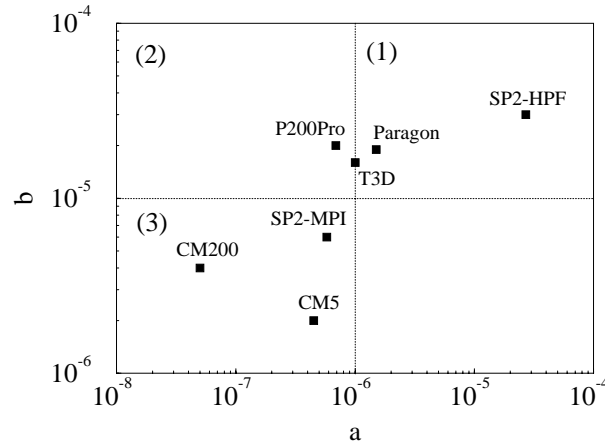


FIGURE: Carte de performance de diverses machines parallèles établie en fonction de leurs coordonnées (a, b) . Les régions distinguent les machines avec (1) les calculs et les communications peu performants, (2) les calculs rapides et les communications lentes et enfin (3) les calculs et les communications rapides.

Quant aux machines à mémoire partagée, nous montrons que ce modèle ne convient pas car il n'y a pas, *stricto sensu*, de communication. Dans ce cas, nous proposons un nouveau modèle, plus "expérimental" comportant également deux paramètres. Un paramètre tient compte de la performance individuelle de chaque

processeur alors que les temps perdus lors de la synchronisation des processus ainsi que lors de la migration des données dans la hiérarchie de la mémoire sont résumés dans un exposant unique.

Propagation des ondes radio en milieu urbain

Notre modèle microscopique pour la propagation d'ondes sur réseau est à la base d'une application complète construite pour les besoins de SWISSCOM. Le but de cette application est de servir à la planification du déploiement d'antennes relais pour les réseaux urbains de téléphonie mobile. Le problème de base est la répartition optimale des cellules de couverture de chaque antenne. Les zones urbaines sont caractérisées par une grande densité des utilisateurs du réseau. Ainsi, la grandeur d'une cellule est limitée par le fait que le nombre d'utilisateurs potentiels présents ne doit pas excéder le nombre de canaux disponibles. C'est pourquoi les antennes sont placées en dessous des toits et leur portée est ainsi fortement influencée par les réflexions et diffractions multiples des ondes se propageant dans l'agencement des rues et des carrefours.

La prédiction de l'étendue d'une cellule étant donné la puissance et la position de l'antenne est un problème difficile dans le milieu urbain. Le problème ne peut pas se résoudre analytiquement et les mesures réelles sont fastidieuses à obtenir. Nous avons développé un outil de simulation par ordinateur rapide basé sur notre modèle d'ondes sur réseaux permettant de calculer la propagation des ondes radio dans l'environnement urbain.

Le processus de simulation commence par construire une représentation discrète bidimensionnelle du quartier contenant la source (l'antenne) dont on veut calculer la portée. Ceci peut être fait simplement en scannant un plan précis; on obtient un domaine de simulation sous forme d'un tableau (par exemple de taille 1000×1000) sur lequel chaque point représente une portion de l'espace réel avec ses caractéristiques. Il faut distinguer quatre types de points: les sites "vide" où l'onde se propage librement, les sites "mur" qui correspondent aux bâtiments et autres obstacles relevant du plan de quartier, le site "source" marquant la position de l'antenne, et finalement les sites "absorbants" encadrant la zone de simulation. Un résultat typique d'une simulation est montré sur la figure de la page suivante.

Chaque site opère une "collision" de ses f_i selon la description générale du modèle donnée plus haut, puis propage ses nouveaux f_i vers les sites voisins. Ainsi rien ne distingue les sites entre eux si ce n'est les coefficients de la matrice de collision. Cependant, avant d'obtenir des prédictions quantitatives fiables il faut encore calibrer et adapter le modèle théorique.

La source représentant l'antenne est définie par ses f_i évoluant indépendamment du voisinage selon une fonction sinusoïdale du temps. L'amplitude

de la source doit être correctement calibrée pour simuler l’effet d’une source d’amplitude unitaire. On obtient un facteur qui dépend notamment de la période de la source. Pour un indice de réfraction donné, homogène sur tout l’espace, la période de la source fixe la longueur d’onde de la simulation. Or, nous avons pu établir que notre modèle de propagation ne peut pas résoudre avec suffisamment de précision des longueurs d’onde inférieures à environ six mailles de réseau. Ainsi l’échelle de discrétisation, c’est-à-dire la taille effective de la maille, fixe-t-elle la longueur d’onde minimale possible. Cette longueur d’onde est toujours beaucoup plus grande que celle utilisée dans la réalité des mesures *in situ* mais nous avons su renormaliser les effets de cette “fausse” longueur d’onde et démontrer qu’elle est bien adaptée aux conditions de notre simulation.

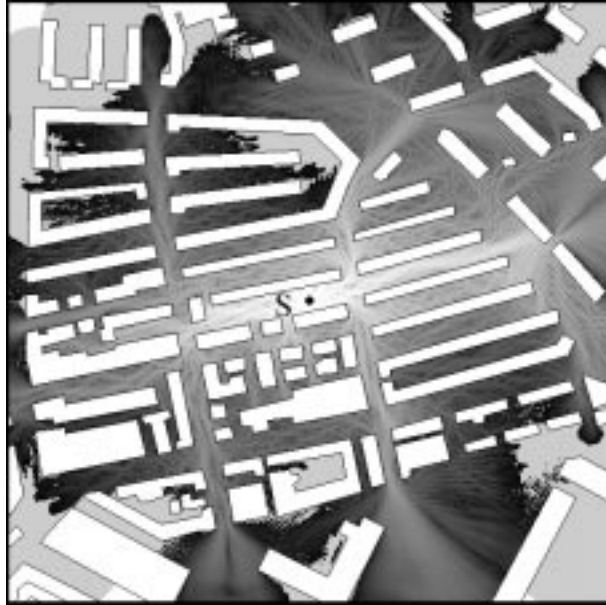


FIGURE: *Résultat d’une simulation pour le cas d’une antenne placée dans un quartier de Berne. Le point S marque l’emplacement de la source, les bâtiments sont bien visibles en blanc alors que l’intensité de l’onde est indiquée par une palette de gris.*

Contrairement à la réalité d’un quartier ou d’une ville, nos domaines de simulation sont naturellement de taille limitée. Il convient d’absorber le plus parfaitement possible les ondes qui atteignent les frontières du domaine si l’on veut simuler un domaine plus étendu. Ce problème est délicat car les ondes inévitablement réfléchies par les bords du domaine peuvent perturber l’ensemble des mesures effectuées à l’intérieur. La solution brutale consistant à annuler tous les f_i sur les bords est mauvaise et nous avons opté pour une absorption graduelle à l’aide de trois couches successives encadrant la zone de simulation. Les sites de ces couches se comportent comme des sites “vide” auxquels on a rajouté un

facteur $\mu < 1$ devant la matrice de collision. Il est ainsi possible de réduire la quantité d'énergie réfléchiée par les bords à moins de 1% de l'énergie incidente.

On considère que l'essentiel du comportement d'une antenne placée en dessous de la limite des toits est capturé par une simulation sur un domaine plan (2D). Cependant, un processus de renormalisation doit tenir compte du fait que, dans la réalité, l'antenne s'apparente plus à une source ponctuelle plongée dans un espace 3D. Nous avons mis au point un procédé original qui permet de tenir compte de ce problème dimensionnel en même temps que celui de la fausse longueur d'onde mentionnée plus haut. L'idée principale de notre renormalisation est que la distance $d(\mathbf{r})$ parcourue par l'onde lors de sa propagation de la source vers le point de mesure \mathbf{r} (mesurée en longueurs d'onde) détermine le taux de décroissance de l'onde. Or, une onde se propageant dans l'espace vide 3D décroît $\propto d^{-1}$ alors que dans un plan vide 2D, elle se comporte $\propto d^{-1/2}$. L'idée est alors de corriger, en chaque point, la valeur simulée par le facteur suivant:

$$\frac{\mathcal{L}_0}{\sqrt{\pi \mathcal{L} d(\mathbf{r})}}$$

où \mathcal{L}_0 est la "vraie" longueur d'onde et \mathcal{L} la longueur d'onde de la simulation.

Nos simulations ont permis de tester notre modèle, notre calibration ainsi que notre méthode de renormalisation sur toute une gamme de situations. Par exemple, le problème de la demi-arrête (dont la solution analytique est connue) démontre que notre méthode tient compte correctement de la diffraction des ondes engendrée par les coins. Dans deux autres situations "réelles", les prédictions du modèle ont pu être quantitativement comparées avec des mesures effectuées en grandeur réelle par SWISSCOM. Il résulte de ces analyses que notre approche est précise, rapide et offre ainsi un nouvel outil fonctionnel de ce domaine clé de l'industrie moderne des télécommunications.

Corps solides et milieux chaotiques

Un des succès les plus remarquables de notre modèle est son aptitude à simuler des phénomènes apparemment très différents sans avoir recours à de grands changements. Ainsi, nous proposons un modèle de corps solide directement issu de notre modèle pour la propagation des ondes. Le "solide" simulé comporte un grand nombre d'atomes et il est capable de se mouvoir dans l'espace en conservant sa structure globale. Les f_i de notre modèle représentent maintenant des distances entre atomes voisins ou, plus précisément, des écarts par rapport à la distance de "repos" entre les atomes du solide. Par conséquent, le mouvement global du solide est engendré très naturellement par la propagation (ondulatoire) de ces perturbations d'un atome à l'autre.

Le solide ainsi défini peut rebondir contre une paroi de façon cohérente (voir figure sur la page suivante) ou encore évoluer dans un champ de force extérieur,

donné par un potentiel gravifique par exemple. Des grandeurs comme la température du solide ou son énergie cinétique sont clairement définies et évoluent de façon cohérente au cours du temps.

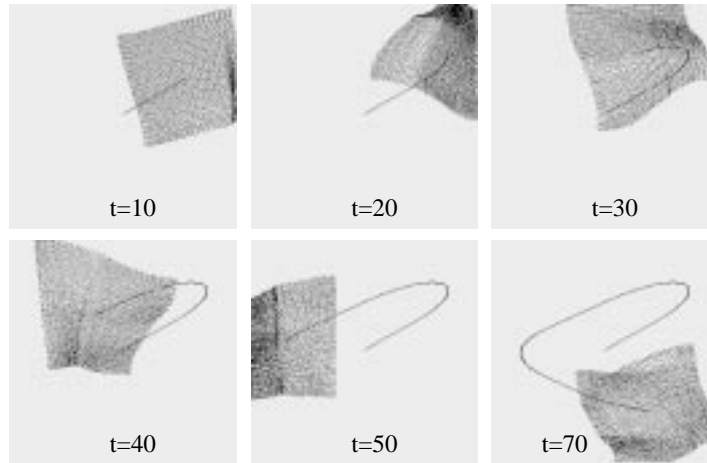


FIGURE: *Mouvement d'un solide déformable formé d'un réseau carré de particules et évoluant dans une boîte rigide selon la règle microscopique de notre modèle d'onde de Boltzmann. La ligne continue représente la trajectoire du centre de gravité.*

L'intérêt principal de notre modèle de corps solide n'est pas la contemplation de sa trajectoire dans une boîte rigide. En effet, une nouvelle interprétation des conditions de bords microscopiques possible (réflexion, absorption, etc.) permet de simuler l'apparition de micro-fractures dans le solide lorsque les contraintes locales dépassent un seuil fixé. Ce modèle permet ainsi d'aborder de façon prometteuse un des problèmes de la physique les plus anciens et les moins bien connus du point de vue fondamental: pourquoi et comment les objets se fracturent-ils ? La puissance et l'originalité de notre approche sont démontrées par la confirmation d'un fait aujourd'hui expérimentalement connu: les fractures se propagent avec une vitesse limite avoisinant la moitié de la vitesse du son dans le solide. Nous avons également pu mettre en évidence les conditions de transition entre une fracture nette et une arborescente.

Enfin, un dernier problème beaucoup plus académique est abordé dans cette thèse: la localisation des ondes dans les milieux chaotiques. Si les inhomogénéités du milieu deviennent denses à l'échelle de la longueur d'onde, l'onde perd peu à peu ses propriétés de propagation et adopte d'abord un comportement plus proche de la diffusion pour finir par être totalement localisée dans l'espace (plus aucune propagation). Grâce à la possibilité de fixer un indice de réfraction différent en chaque site, notre approche se révèle bien adaptée à l'étude de ce problème.

Nous avons pu mettre clairement en évidence des comportements diffusifs et sous-diffusifs en variant la densité et l'intensité des diffracteurs microscopiques (sites ayant un indice de réfraction supérieur à celui du milieu ambiant).

Conclusion

Les résultats obtenus lors du développement et des applications de notre modèle discret pour la propagation ondulatoire ouvrent de nombreuses perspectives de recherche. Notre approche a réussi à ne conserver que l'essentiel des phénomènes étudiés et à les traduire sous une forme particulièrement simple et adaptée à la simulation par ordinateur. Nous avons démontré la généralité de notre modèle par la variété des phénomènes qu'il peut simuler. Sans doute, notre modèle discret partage-t-il un peu de cette universalité caractéristique des ondes et la résolution d'autres problèmes ne manquera pas de profiter des expériences et des résultats présentés dans cette thèse.

Contents

1	Models, Simulation and Computers	1
1.1	Simulations versus Resolutions	1
1.2	Examples of CA Simulations	4
1.3	Cellular Automata and Statistical Physics	7
2	Lattice Boltzmann Waves	11
2.1	Derivation of the Lattice Wave Model	14
2.1.1	Continuity equations for I and \vec{J}	18
2.1.2	The stress tensor Π	21
2.1.3	The wave equation	23
2.1.4	The time reversal invariance, viscosity	25
2.1.5	Dissipation	26
2.2	Transmission Line Matrix	30
2.2.1	The symmetric 5×5 matrix formulation	30
2.2.2	The dispersion relation	33
2.2.3	TLM versus the finite difference scheme	36
2.3	Generalization of the linear approach	37
3	Algorithms & Implementations	43
3.1	Introduction	43
3.2	LB Wave Automata Kernel: A New Benchmark	44
3.3	Parallelization	50
3.3.1	Data Parallel Programming Model	51
3.3.2	HPF, The New Standard	53
3.3.3	Message Passing Programming Model	53
3.3.4	Shared Memory Programming Model	57
3.4	Scalability	60
4	Simulation of Radio Waves Propagation	63
4.1	The numerical model	64

4.1.1	Point Sources	66
4.1.2	Sites of Reflection	67
4.1.3	Sites of attenuation	68
4.2	The Renormalization Method	70
4.3	The Simulated Wave Length “Problem”	73
4.4	Huygens Principle	74
4.5	Optimizing the Algorithm	75
4.5.1	The Sector-shaped Decomposition Model	75
4.5.2	The Ring-shaped Decomposition Model	76
4.6	Simulations: Calibration & Predictions	78
4.6.1	The Knife-wedge Test Problem	79
4.6.2	The Four-corner calibration	81
4.6.3	A Full Urban micro-cells	82
5	Other Applications	85
5.1	Large-scale Moving and Cracking Objects	85
5.1.1	The 1-Dimensional CA String Model	86
5.1.2	The 2-Dimensional Sheet model	90
5.1.3	Solid Body Motion	93
5.1.4	Fracture process	98
5.2	Wave Localization	103
6	Conclusion	111
A	The Matrix Warehouse	115
A.1	The TLM-matrix	115
A.2	The <i>Parflow</i> -matrix or the (c, c) -matrix	115
A.3	The asymmetric (a, N, \mathcal{N}) -matrix	116
A.4	The symmetric (n) -matrix	116
A.5	The symmetric (n, ξ) -matrix	117
A.6	The 1-dimensional matrix	117
A.7	The diffusion matrix	117
B	The PVM Code of the Model	119
C	A <i>parflow</i> example	131
C.1	introduction	131
C.2	The parameters of the simulation	131
C.3	The results of the simulation	134
	Bibliography	135
	Index	141

List of Figures

1.1	Example of Liesegang's bands	5
1.2	Discrete diffusion and continuous-like deposition	6
1.3	Homogeneous steady configuration of CA competing cells	6
1.4	CA model for car traffic in a Manhattan-like city	7
2.1	Different possible lattices in 2-dimensional space	15
2.2	Illustration of the discrete automaton evolution	16
2.3	Focusing effect of a lens with different refraction indexes	24
2.4	Focusing effect of a parabolic-shaped mirror	30
2.5	Contour lines of the dispersion relation	34
3.1	Simplified algorithm of LB wave automaton	45
3.2	Memory representation of the automata	48
3.3	Sequential Performance on different machines	49
3.4	Comparison between C, F77 and F90	50
3.5	Performance analysis for data parallelism (CM 200)	52
3.6	Performance analysis for HPF (IBM sp2)	54
3.7	Message-passing implementation	55
3.8	Performance analysis for message passing (SP2)	55
3.9	Performance analysis for message passing programming model	56
3.10	Performance analysis for a shared memory machine (Origin 200)	58
3.11	Performance map for parallel machines	60
4.1	The γ factor as a function of T	67
4.2	Aspect of the f 's issued from a source	68
4.3	Effective reflection coefficient in function of μ	69
4.4	Total anisotropic factor for renormalized point source	72
4.5	Evolution of the causality sub-domain	76
4.6	Dynamic supersonic ring-shaped domain	77
4.7	Synoptic representation of the simulation parameters	79
4.8	Knife wedge test problem	80

4.9	The four-corner problem (Fribourg)	81
4.10	Result of the four-corner problem	82
4.11	The complete Urban microcell (Bern)	83
4.12	Result in the complete Urban microcell	84
5.1	Motion of a 1-dimensional string	87
5.2	A 2-dimensional checkerboard solid	90
5.3	Motion of a deformable 2-dimensional LB object	94
5.4	Reflection of the solid against a wall	95
5.5	Motion of the 2-dimensional solid	96
5.6	Energies exchange for solid evolution	97
5.7	Trajectory of a solid in a gravitational potential	98
5.8	Energies exchange for the solid in a gravitational potential	99
5.9	Contour plot of the energy E in a mode I-loaded sheet	101
5.10	Crack speed and instabilities in fracture dynamics	103
5.11	Energy propagation pattern in a random medium	107
5.12	Transition to localization of an impulse in a random media	108
5.13	An approach to strong localization	109

Models, Simulation and Computers

Computers now play a dominant role in the scientific process and eventually progress. Their increasing power used to simulate the reality, not only allow us to get a deeper understanding of complex phenomena but is about to provide us with a completely new type of environment, the so-called virtual reality. Virtual reality is nothing more than an extremely evolved kind of model, speaking more likely to our feelings than to our reason.

1.1 Simulations versus Resolutions

Letting the virtual reality to the community it belongs to, what is the necessity behind the common effort to build up models of our physical or socio-economical surrounding world? Undoubtedly, this is not an easy question and we believe that it contains the very motivation of the present work which address an extensive, but by no ways definitive study of a particular numerical model for waves. Wave propagation is a kind of universal phenomenon which may be observed throughout the nature, from everyday's experiences up to sophisticated nanoscopic analysis. Consequently, our wave model model will be able to address apparently unconnected problems like radio wave propagation in urban area and brittle fracture dynamics. Before going in the details of the derivation of our model and the various aspects of its application, this first chapter will go through some general considerations in order to situate our work in the successive context of science, physics and computers.

A model of the reality, be it from any kind, is the ultimate expression of our

rational understanding. Science in general may be seen as a more or less coherent collection or succession of permanently temporary models whose simplicity and accuracy must reflect the smartness of the scientist's vision. Indeed, the only possible certitude about them concern their refutation in case of inadequacy with the observed reality. In other words, scientific works should, from our point of view, implicitly contain this preamble of modesty and honesty: "Everything happens as if...". We believe it is impossible to be more peremptory!

Models are not only the most synthetic outcome of a successful research but they also constitute a powerful investigation tool in the process of highlighting the most important underlining mechanisms of complex phenomena. In that sense we may say that scientific investigations suppose to intensively play (and replay), however a serious game, with models or in other words, to make simulation. The rule of the game is well known: "Don't cheat with observations" and the winner is who is able to make accurate predictions based on his understanding.

Similar to the different levels of programming languages available on a computer, there is different level of models which may be distinguished through the nature of their ingredients. The lowest level model is the experimental or say, the physical model. Every laboratory experiment is allowed to enter this category. The ingredients of the experimental model are so close to the reality - for instance you will use "real" stones to check the gravity in your lab - that they are often considered as the relevant reality itself. However, the availability of such low-level, or quasi-real, models are fundamental for the progress of science since they often provides the only definitive way to refute a higher level model. It turns out that scientific domains for which laboratory's experiments are momentarily or essentially impossible (this is case if the associated phenomena are not reproducible, like in the fields of cosmology or paleontology for instance) are far more speculative and allow coexistence of contradictory conclusions. If they must rely exclusively on direct observations, scientists often lack of "smoking guns" confirmations or refutations of their statements.

Careful observations or intensive experiments may be carried out by everybody but this is by far not enough to explain today's scientific progress. Modern physics, namely dating back at least to Sir Isaac Newton's *Philosophiae naturalis principia mathematica* published in 1687, usually derives mathematical models from experiments or direct observations. The mathematical models we are talking about are extreme high-levels models whose ingredients actually are abstract variables expressing physical laws by means of differential equations. Indeed, the decisive breakthrough in differential/integral calculus methods is intimately linked with Newton's formulation of his three laws of motion. Newton applied these laws to Kepler's laws of orbital motion formulated by the German astronomer Johannes Kepler and derived the law of universal gravitation, perhaps the most challenging physical problem of that late seventeenth century.

The immediate success of the method using differential equations to model physical phenomena in a tractable form soon convinced the remaining former

scientific community. This thought process was generalized and founded other modern sciences like chemistry or later, biology, all participating in the formidable technological and intellectual advance of our twentieth century. However, the abstract mathematical model is not exactly a panacea for the difficult task to understand our surrounding world. Two kinds of difficulties may be pointed out which we dare to characterize with some arbitrariness as external and internal limitations of the differential equations approach.

External limitations becomes more obvious if we focus on the profusion of complex questions emerging from contemporary domains like economy, sociology, psychology or even politics. The issue of these questions seems so far to defy any global mathematical formulation even if numerous attempts have been made. These problems are of fundamental interest for the human society and they have been addressed with a pre-mathematical approach using common language's logic or profession of faith. We believe this would be nonsense to make a trial against this approach, firstly because most of our everyday life relies on it and secondly because most of the scientific intellectual reflex is still characterized by language's logic or even profession of faith[1].

Even applied in its favorite domains like physics, a mathematical model is sure to show up some intrinsic, or intern dramatic limitations. Indeed, it turns out that sooner or later, the equations becomes so complicate that any resolution lies beyond an analytical approach. If the equations may not be solved, the model is loosing its power to express our mental vision of the reality, its efficiency as a tool to develop and test our comprehension of a phenomenon and last but not least its capability to provide predictions, which is definitively the final goal of the whole effort. Thus differential equations must be solved to make sense as mathematical models of the reality and we believe this approach was giving a second life with the advent of electronic computers allowing new numerical solutions.

Consequently, it is not a matter of accident if the very origin of electronic computers is dating as back as the 1940's when major physicists were involved in the atomic program, one of the most challenging physical question of that war time. In this background, the purpose of the computers was to make calculations, namely to provide numerical solutions to the physicist's equations. Strictly speaking, those computers do not provide any new kind of model, they are not actually simulating the phenomenon under study but are rather simulating the mathematical treatments required to derive solutions from a given mathematical abstract model. During the past fifty years a great deal of effort was devoted to develop fast, accurate and stable numerical methods and today's use of computers in science is still largely dedicated to the resolutions of complicated equations.

However it turns out that computers not only offer a precious help to scientist's minds facing the difficult task to deal with their equations but they also provide a completely new class of model for making direct numerical simulation of our world. The idea is to design a synthetic model of universe in which the physical laws are expressed in terms of simple local rules on a discrete space-time and

-phase structure. Although invented in a more restrictive context, today *Cellular automata models* refer best to this kind of approach. The pioneer of cellular automata computing is certainly John von Neumann who, at the end of the forties, was involved in the design of the first digital computers. He was interested, and he had succeeded in finding a logical, discrete abstraction of the self-reproduction mechanism without any direct reference to the biological process involved. Our present work is a direct continuation of the simulation approach's history initiated by Von Neumann. This is a successful story for natural science, intimately linked with the development of computers, but also a promising perspective for other fields like what Galam[2, 3] called social physics.

Thus, cellular automata have been an early attempt to design artificial life and can certainly be still exploited to progress in this field. In a related spirit, the mathematician John Conway proposed in 1970 his, now famous, *game of life*. Later on, in the beginning of the eighties, Stephen Wolfram has studied a family of simple one-dimensional cellular automata rules: *Wolfram's rules*. These successive developments brought to a wide audience that cellular automata are discrete dynamical systems able to exhibit very complex or collective behaviors like those observed in the continuum but in a dramatically simpler framework.

The use of cellular automata rule as a model of real, physical phenomena like fluid flows, was given much thoughts in the eighties, after it was recognized that the so called HPP and FHP[4, 5] lattice gas model was in fact a particular class of cellular automata. From that time on it is considered as an effective numerical tool which is able to retain important aspects of the microscopic laws of physics like, for instance, simultaneity of motion, locality of interactions and fundamental symmetries. However, contrary to the first expectations, lattice gas models of fluids have not been able to surpass more traditional numerical method applied to the resolution of hydrodynamics equations. But lattice gas automata have been much more successful to model complex situations for which traditional computing techniques are hardly applicable and the reader may refer to [6] to appreciate the state-of-the-art in cellular automata modeling of physical systems. The next section briefly brings up various models studied by the authors, but not discussed in this thesis.

1.2 Examples of CA Simulations

For example, a very interesting class of problems studied by the Bastien Chopard, Pascal O. Luthi and Michel Droz, in a preceding work[8, 7, 9] concerns the so-called Liesegang patterns formation[10]. This problem is of the class for which a description in terms of rate equations is not sufficient. It turns out that some aspects of the fluctuating microscopy is essential so that first-principle analytical results are scarce. Liesegang patterns are produced by precipitation in the

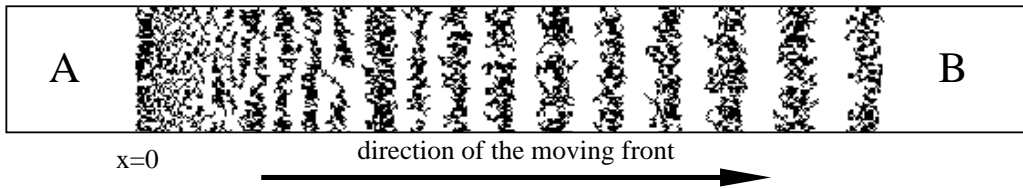


FIGURE 1.1: Example of Liesegang bands as obtained from a simulation of our cellular automata model; see [7]. The CA represents a test tube initially filled with B particles. Particles A are introduced at $x = 0$ and diffuse inside the test tube. Particles A and B react, precipitate and aggregate in alternate bands, in the wake of the moving reaction front.

wake of a moving reaction front. Many experiments exhibiting such a pattern formation consist of a test tube containing a gel in which a chemical species B is uniformly distributed with a given concentration. Another species A is allowed to diffuse into the tube from its open extremity and chemically react with B. As the reaction goes on, formation of *a priori* unexpected consecutive bands of precipitate is observed in the tube; see figure 1.1. Contrasting a hundred of years of various investigations, our approach was based on a microscopic model in which the essential features of the kinetics were modeled in a very simple way. A unique local rule accounts for the various aspects of the problem: diffusion of the species, virtual chemical reaction, nucleation and aggregation process of the precipitate. The model allowed large scale numerical simulations which provided clear verifications of different macroscopic laws usually describing this phenomena.

Another example illustrating the power of cellular automata based numerical model is given by the study of the role of diffusion in irreversible deposition [11]. The often irreversible adsorption of proteins and other particles at the solid-liquid interface is a widespread process of fundamental importance in nature. This non-trivial problem including the transport of the particle from the bulk to the proximity of the surface and the subsequent adsorption onto the surface was investigated at a virtual microscopic level using a cellular automata model for particle diffusion and adsorption, see figure 1.2. Numerical simulations allowed the study of the initial regime, which determines the subsequent evolution and during which the fluctuating particle flux toward the surface is determinant. The model highlights the effect of the microscopic details of the diffusive motion on the kinetics of the deposition and the jamming limit of the coverage. Our approach permit to understand the limit and the accuracy of a more simple model, the so-called random sequential adsorption (RSA) approach of the problem.

Our third example addresses a fundamental biological problem. A cellular automaton is constructed for modeling the formation of the central nervous system of a living organism like the *Drosophila embryo*[12] *Drosophila*. This is an experimentally well studied system in which complex interactions between

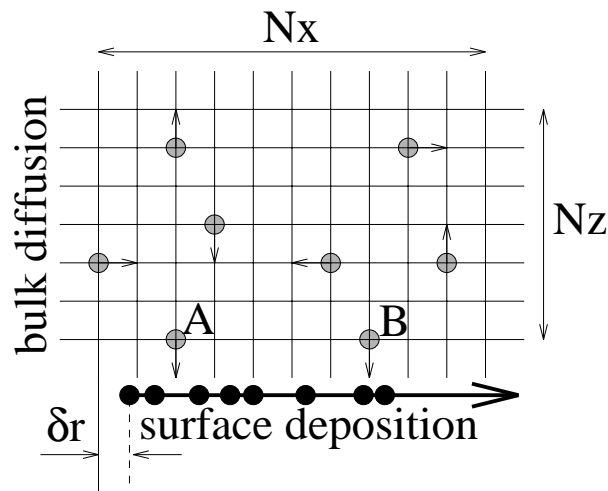


FIGURE 1.2: Discrete diffusion on the lattice and continuum deposition on the surface. δ_r is randomly chosen at each time step. The deposition is accepted only if there is no resulting overlap (case A), otherwise the particle is reflected (case B); see [11]

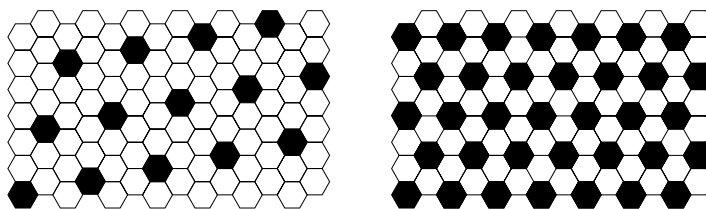


FIGURE 1.3: The two theoretical “peaceful” configurations resulting from a competition between nearest neighbors on a hexagonal lattice, irrespective of any detail. Black cells represent “winners” or future neuronal cells, and white cells represent “losers”. On the left, the sparsest possible packing of winners without emerging conflict ($1/7$ of the sites). On the right is the closest packing of winners without conflict ($1/3$ of the sites).

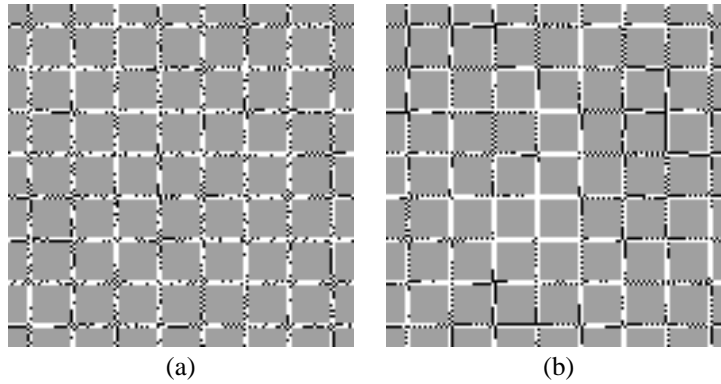


FIGURE 1.4: *Traffic configuration after 600 iterations, for a car density of 30%. Streets are white, buildings grey and the black pixels represent the cars. The situation (a) corresponds to an equally likely behavior at each rotary junctions, whereas image (b) mimics the presence of traffic lights. In the second case, queues are more likely to form and the global mobility is less than in the first case.*

neighboring cells appears to drive their differentiation into different types. It has been proposed that all cells initially have the potential to become neuronal cells, and all thrive to do so, but those which differentiate first block their as yet undifferentiated neighbors from doing so. The cellular automaton makes use of observational evidence for lateral inhibition mechanism involving signaling products of the neuralizing genes. The key idea of the model that of a competition between cells. Cells are continuously producing the signal, but the production rate is lowered by the inhibitory signals received from neighboring cells which have advanced further along the developmental pathway. Comparison with experimental data have shown that it well accounts for the observed proportion (25%) of future neuronal cells. The final configuration of winners is a subtle mixture of the two extreme configurations shown in figure 1.3.

We have also proposed a road traffic CA model suitable to simulate an entire urban environment [13]. We considered the traffic in a Manhattan-like city (see figure 1.4) at a discrete microscopic level: a cell represent a portion of street that may be free or occupied by an individual car. Thus we could study the complex and non-uniform dynamics of the length of the car queues. The street length between two junctions and the turning strategies at rotaries turns out to be relevant parameters of the model.

1.3 Cellular Automata and Statistical Physics

The previous examples, as well as numerous other cellular automata designed during the past twenty years, make it clear that cellular automata may have strong

similarities with real complex systems, despite the simplicity of the interactions they take into account. If not intuitive, there is however nothing exceptional in the fact that a dramatically simplified discrete microscopic scheme is eventually able to behave like a real, and far more complex system. This comes from the generally complex relationships existing between the microscopic, mesoscopic and macroscopic worlds. The length- and time-scales of observation completely drive our assessment of the reality. Often, the macroscopic behavior of a system composed of many interacting “atomic” constituents depends very little on the microscopic details of the interactions. This is sometimes so evident that we may eventually speak of completely different, weakly connected, levels of reality each of them with its own complexity. Thus, on the one hand we may observe simple macroscopic behavior out of numerous very complicated microscopic interactions and on the other hand, complex collective behavior may emerge out of very simple microscopic assumptions. If the former statement is the corner stone of the thermodynamics and statistical mechanics approach, the later constitutes the foundations of our present simulation approach.

Thus, statistical mechanics and cellular automata models both address the question of the link between the microscopic and the macroscopic aspects of the reality, however with a different point of view. The thermodynamic approach describes macroscopic objects and their equilibrium properties resulting from various thermal and mechanical energy exchange. This description uses few macroscopic variables such as temperature, pressure, etc. and fully ignores the microscopic nature. Indeed, some results and principles of thermodynamic were historically established by scientists rejecting the atomic approach of matter. Describing a macroscopic system at a microscopic scale involves so many degrees of freedom that they cannot be treated individually. Moreover, even if it were possible, there is actually no or little interest in knowing that much information. What is interesting in the study of complex systems is the apparition of a collective, or coherent, behavior. An enlightening example of collective behavior emerging out of some microscopic disorder is the propagation of sound waves in the air. Considering individual degrees of freedom as being part of a more global ensemble which can be described by collective variables is the point of statistical mechanics. Moreover, macroscopic system are often out of equilibrium and the study of the dynamics of collective behaviors based on fundamental principles like particle interactions is a very difficult task where approximations and/or simulations are needed most of the time.

The most simple approach is the mean-field approximation where each particle is supposed to evolve in a potential created by all the other particles. This type of approximation is able to convert an unsolvable \mathcal{N} -body problem in \mathcal{N} more handleable 1-body problems. This approach may be improved by taking into account some simplified correlations between the particles, like the collisions between two of them, for instance. This is actually done by the Boltzmann equation, a closed, kinetic equation for the one body distribution function $f(\vec{x}, \vec{v}, t)$. f

is defined in the 6-dimensional position-velocity phase space so that $f d\vec{x} d\vec{v}$ represents the average number of particles contained in the phase space elementary volume $d\vec{x} d\vec{v}$ centered at \vec{x} and \vec{v} . The distribution function f allows us to derive all the macroscopic variables, and the purpose of the Boltzmann equation is to express time evolution in a simplified context. It is a non-linear integro-differential balance equation which reads, in a compact form [14],

$$\frac{d}{dt} f(\vec{x}(t), \vec{v}(t), t) = \Leftrightarrow I^{(-)}(f) + I^{(+)}(f)$$

It simply means that the net temporal change in the distribution function f due to two-body collisions is the sum of a gain $I^{(+)}$ and loss $I^{(-)}$ term established on microscopic, often quantum-mechanical considerations. These terms I are triple integrals over three velocity spaces: the income velocity of the other particle and the outcome velocities of the two colliding particles. This Boltzmann equation cannot be solved in the general case and the key idea behind most of the approximation technique is that it describes the return of the system to an equilibrium state. Providing that the perturbation is small, an idea is to linearize the left-hand side. Moreover, the usual hypothesis are based on the concept of a possible local equilibrium, which supposes that the equilibrium in the velocity space is obtained much before the equilibrium in the position space. The Boltzmann equation then transforms to

$$\frac{d}{dt} f(\vec{x}(t), \vec{v}(t), t) = \frac{f^{(0)} \Leftrightarrow f}{\xi}$$

where ξ is a relaxation time and $f^{(0)}(\vec{x}, \vec{v}, t)$ the local equilibrium distribution function. The point is that $f^{(0)}$ depends on \vec{x} and t only through some macroscopic variable defining an equilibrium like temperature, for instance.

The approach adopted in the present work to address the problem of wave propagation in strongly heterogeneous media is a generalized cellular automata approach called a *lattice Boltzmann model*. It is a generalized cellular automata model because instead of having a boolean dynamic such as that required by a strict (historical) definition, we shall work on real numbers representing, somehow, the probability for a cell to have a given state. This method was invented at the end of the eighties, by McNamara and Zanetti [15], Higuera, Jimenez and Succi [16] to overcome the statistical noise and the little flexibility to adjust parameters related to the fully discrete nature of the boolean cellular automata. It is called a lattice Boltzmann model because the rule of the automata will take the form of a fully discrete, in position, velocity and time, Boltzmann equation. The thought process is however quite different: to infer a powerful cellular automata model we must foresee the very essential microscopic features of given phenomena and translate them in a suitable form to get an effective numerical tool. To this end it is acceptable and even beneficial to start with an over-simplified, let's say a

little realistic microscopic world providing it will show up the right behavior once observed at the appropriate scale. Making an intense use of a lattice Boltzmann on today's parallel computers offers a powerful and promising tool for the study of complex dynamical systems where other analytical approaches fail to give an accurate and tractable picture of what happens.

The purpose of the remaining part of this thesis is to find a way to represent waves, namely a kind of universal phenomena in nature, in a way suitable to a (parallel) computer processing. The method adopted will allow fast and accurate computer simulation of discrete waves and resulting in new understandings of complex real phenomena. Chapter 2 presents an original and accurate derivation of our lattice wave model to show up how it can be used to model scalar wave propagation in complicated medium. After this rather mathematical approach we will present, in a chapter 3, the details of a general purpose algorithm using this method and demonstrate the ease and the efficiency of its implementation on a wide range of different computers ranging from a simple personal computer up to large parallel supercomputers. Chapter 4 goes through the most successful application of our method in the context of the difficult problem of radio wave propagation in urban areas. Finally and before concluding, chapter 5 will demonstrate the universality of our approach by presenting promising, apparently completely different applications addressing the challenging problems of the dynamics of fracture propagations in brittle solid and the more academic-like problem of wave localization in random media.

Lattice Boltzmann Waves

Waves are everywhere around us and may be observed at all length and time scales. Water waves are perpetually traveling the ocean and occasionally manifest impressive destructive energy as they break on the shore; they are undoubtedly at the very origin of our conception of waves of any kind and are their most concrete manifestation. Sound and light waves are so intimately linked with our perception of our immediate surrounding that it is hard to stay aware of their presence and meaning. Electromagnetic waves on their own are responsible for all our radio and telecommunication means and finally the very deep nature and stability of matter rely on the quantum waves associated with electrons and atoms. Wave phenomena clearly occupy a central position in our study of the physical world.

Waves are so universal that they must reflect some basic symmetries of a dynamical system rather than any other particular details. Indeed, waves are the way to transport energy or say, whatever information, in a proper manner through a medium with the particularity that the medium itself undergoes no global motion. What is meant here, is that the speed at which a wave can carry information is finite, in other words, wave transport has the basic property of causality stating that consequences must necessarily succeed causes. This property is directly related to the phase information carried by the wave, namely its oscillating appearance. Diffusion phenomena, like the propagation of temperature in a solid, is a clear-cut counterexample: no oscillations and an infinitely fast propagation of boundary conditions are assumed by the governing diffusion equation.

Mathematically, the classical wave propagation of a scalar quantity, ψ is expressed by the so-called wave equation:

$$\partial_t^2 \psi \Leftrightarrow c^2 \nabla^2 \psi = 0$$

where c is the wave speed. This second order equation itself may not be considered as a meaningful physical equation. Indeed, it does not tell much on the very nature of the underlying process except the presence of linearity and time reversal invariance. As a matter of fact, changing t by $\Leftrightarrow t$ leaves the equation unchanged and if ψ_1 and ψ_2 are solutions then so is $\psi_1 + \psi_2$.

The ingredient of the underlying physical process must be contained in a more fundamental set of equations. At this usually “microscopic” level, it turns out that physics is usually described by first order equations as for instance, is the case of the Hamiltonian formalism with the set of canonical equations. These basic equations determines uniquely the evolution of the system providing its state is precisely known at one particular instant of time. As we shall see through some examples, different sets of first order physical equations may result in wave equation and a paradigm will emerge which will inspire us for the design of our discrete wave automaton.

Sound waves is the propagation of small perturbations in a fluid initially at rest. Suppose that the pressure p and density ρ are always close to their equilibrium value p_s and ρ_s and all flow speeds are small; this is the domain of linear acoustic. The propagation of the first order small pressure perturbation $p_a = p \Leftrightarrow p_s$ (similarly $\rho_a = \rho \Leftrightarrow \rho_s$) is governed by a wave equation which is easily derived from the three first-order equations given below; see [17]. A thermodynamic equation which follows from the supposition that there is no heat exchange during the process and that the dynamics is time reversible, and two other equations are continuity equations expressing conservation laws:

$$\begin{aligned} \rho_a &= \kappa \rho_s p_a && \text{Thermodynamic equation} \\ \partial_t \rho_a + \rho_s \nabla \vec{v} &= 0 && \text{Mass conservation} \\ \rho_s \partial_t \vec{v} + \nabla p_a &= 0 && \text{Momentum conservation} \end{aligned}$$

To obtain a wave equation for ρ_a one must firstly take the time derivative of the mass conservation law. Then, one takes the divergence of the momentum conservation law to replace the term $\partial_t \nabla \vec{v}$ of previous equation. Finally, the term $\nabla^2 p_a$ is re-written in terms of ρ_a using the thermodynamic equation, and we obtain

$$\partial_t^2 \rho_a \Leftrightarrow \frac{1}{\kappa \rho_s} \nabla^2 \rho_a = 0$$

As for electromagnetic or light waves in vacuum, they are a direct consequence of the Maxwell equations. A wave equation for each component of the electric field \vec{E} may be also easily derived from the three first order equations given below, following a comparable method as given above. The Gauss’s law is the consequence, or the differential form of the Coulomb’s law which states both the linearity or the additivity of electrostatic forces and the inverse square law for its magnitude. The two electrodynamic equations have not the form of conservation

laws and this exhibits the profound differences that may exist between two kinds of waves.

$$\begin{aligned}\nabla \vec{E} &= 0 && \text{Gauss's law} \\ \frac{1}{c} \partial_t \vec{E} \Leftrightarrow \nabla \times \vec{B} &= 0 && \text{Ampère's law} \\ \frac{1}{c} \partial_t \vec{B} + \nabla \times \vec{E} &= 0 && \text{Faraday's law}\end{aligned}$$

Our last example is directly derived from the Maxwell equations but takes a form rather similar to fluid equations given above. It concerns the wave propagation of the energy density $u = (\vec{E}^2 + \vec{B}^2)/8\pi$. Following [18] we write the conservation of the energy and momentum of electromagnetic fields in the vacuum, using the definition of the Poynting vector $\vec{S} = c(\vec{E} \times \vec{B})/4\pi$, as follows:

$$\begin{aligned}T_{\alpha\beta} &= \Leftrightarrow u \delta_{\alpha\beta} + \frac{1}{4\pi} (E_\alpha E_\beta + B_\alpha B_\beta) \\ \partial_t u + \partial_\alpha S_\alpha &= 0 && \text{Poynting theorem} \\ \partial_t \frac{S_\beta}{c^2} + \partial_\alpha T_{\alpha\beta} &= 0 && \text{Conservation of momentum}\end{aligned}$$

where $T_{\alpha\beta}$ is called the Maxwell stress tensor. We have introduced some notations that will be used throughout this chapter: Greek indices account for the space dimensions and summation is assumed if repeated. For example:

$$\partial_\alpha T_{\alpha\beta} \doteq \sum_{\alpha=x,y,z} \partial_\alpha T_{\alpha\beta}.$$

Because \vec{E} and \vec{B} are divergence free in vacuum, the second term in the definition of the Maxwell stress tensor vanished once plugged in the equation for the momentum conservation and we obtain again, a wave equation for the energy density u .

The different nature of the three examples given above illustrates well the various possible wave phenomena. In this chapter, we will derive a fully discrete model for wave propagation based upon a paradigm inspired by the examples. Indeed, the particularity of our model is that it is not a direct discretisation of the wave equation but rather the representation of a well defined virtual process. The ingredients, or the arguments retained for our paradigm are the conservation law for two quantities, one scalar and its associated current and an additional, constitutive equation which closes the system. The constitutive equation encapsulates the physics of the medium response, thus basically linearity and time reversal invariance. Written with the variable names used in the remaining of this chapter, we have

$$\Pi_{\alpha\beta} = I\delta_{\alpha\beta} \quad (2.1)$$

$$\partial_t I + \partial_\alpha J_\alpha = 0 \quad \text{Conservation of } I \quad (2.2)$$

$$\partial_t J_\beta + \partial_\alpha \Pi_{\alpha\beta} = 0 \quad \text{Conservation of } J_\beta \quad (2.3)$$

Combining equations (2.1), (2.2) and (2.3) gives finally a wave equation for I :

$$\begin{aligned} \partial_t(2.2) &\rightarrow \partial_t^2 I + \partial_t \partial_\alpha J_\alpha = 0 \\ \partial_\beta(2.3) &\rightarrow \partial_\beta \partial_t J_\beta + \partial_\alpha \partial_\beta \Pi_{\alpha\beta} = 0 \\ (2.4) \text{ in } (2.4) &\rightarrow \partial_t^2 I \Leftrightarrow \partial_\alpha \partial_\beta \Pi_{\alpha\beta} = 0 \\ \text{Finally, with (2.1)} &\rightarrow \partial_t^2 I \Leftrightarrow \nabla^2 I = 0 \end{aligned} \quad (2.4)$$

For convenience we will name I a generalized charge density and J_α a generalized current density but it may well represent other quantities like for instance an excess of mass and momentum density, respectively.

The remaining part of this chapter is organized as followed: section 2.1 makes the derivation of the our Lattice wave model, next, section 2.2 will focus on the so-called transmission line matrix formulation of the model and derive its dispersion relation for comparison between our model and the direct discretisation. Finally, in section 2.3, some hints are given concerning the possible generalization of the formalism.

2.1 Derivation of the Lattice Wave Model

A wave phenomena in a discrete space-time universe (note that we are not talking about a discretisation of the wave phenomena) can be thought of as being mediated by a set of perturbation traveling along a lattice. Thus, during a time step, a local perturbation propagates towards the nearest neighbors to a rule yet to be determined. The model that will be developed hereafter is a generalized cellular automata. This means that it is fully discrete in space and time but the state of a cell is defined by a set of real-valued numbers. Consequently, the number of possible states for a cell is not strictly finite or at least, it is as finite as the number of different floating-point values available on a computer. The model which can be assimilated with a Lattice Boltzmann model, is defined on a regular lattice occupying a \mathcal{N} -dimensional space. Each lattice site is connected to N of its neighbors; N is named coordination number. The links are labeled by \vec{v}_i as shown for different 2-dimensional examples in figure 2.1.

The state of the automaton is defined by the set of real-valued quantities f_i 's traveling along the lattice directions \vec{v}_i and an additional rest flux f_0 (i.e. $\vec{v}_0 \doteq 0$)

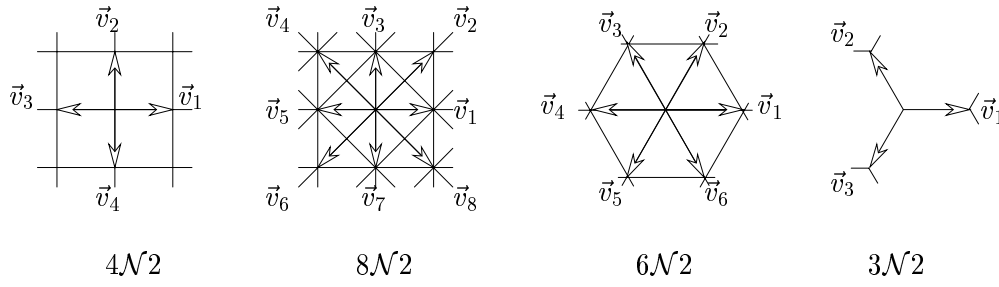


FIGURE 2.1: Different possible lattices and lattice coordination number in 2-dimensional space with their associated label. $4\mathcal{N}2$ means 4 directions and 2 dimensions.

at each lattice site \vec{r} . The dynamics of the automaton is synchronized through all lattice site. A discrete time step of evolution may be formally decomposed in two phases, collision and free displacement. The collision results in building up the new outgoing f_i 's as a function of the old entering f_i 's. The flux are then moving synchronously on the lattice according to a discrete clock of time step τ and the associated speed \vec{v}_i . The \vec{v}_i modulus are such that, in one time step, the flux travel on lattice spacing exactly; see figure 2.2 for an illustration. In other words an outgoing flux at some time t will transform in some in-going flux at a time step later. More precisely, the dynamics of our model is given by a formula like:

$$f_i(t + \tau, \vec{r} + \tau\vec{v}_i) \Leftrightarrow f_i(t, \vec{r}) = \Omega_i(\mathbf{f}(t, \vec{r})) \quad (2.5)$$

where $\mathbf{f} = (f_0, f_1, f_2, \dots, f_N)$. The meaning of the collision term Ω is easy to understand: it encapsulates all the details of the model and if $\Omega = 0$ for instance, the f_i 's are traveling freely or unperturbated across the lattice. Now suppose a certain state of equilibrium exists locally for the dynamics encapsulated in Ω . A state $\mathbf{f}^{(0)}$ of local equilibrium means that we have

$$f_i^{(0)}(t + \tau, \vec{r} + \tau\vec{v}_i) = f_i^{(0)}(t, \vec{r})$$

This means that $\mathbf{f}^{(0)}$ is solution of

$$\Omega_i(\mathbf{f}(t, \vec{r})) = 0$$

A Boltzmann equation is supposed to describe a system relaxing towards equilibrium. Consequently we may rewrite (2.5) with the expression of the equilibrium $\mathbf{f}^{(0)}$ and the relaxation time ξ :

$$f_i(t + \tau, \vec{r} + \tau\vec{v}_i) \Leftrightarrow f_i(t, \vec{r}) = \frac{1}{\xi} (f_i^{(0)}(t, \vec{r}) \Leftrightarrow f_i(t, \vec{r})) \quad (2.6)$$

This equation is still very general but the interesting point is that $f^{(0)}$ may be appropriately chosen so as to produce a given behavior. We will take advantage of

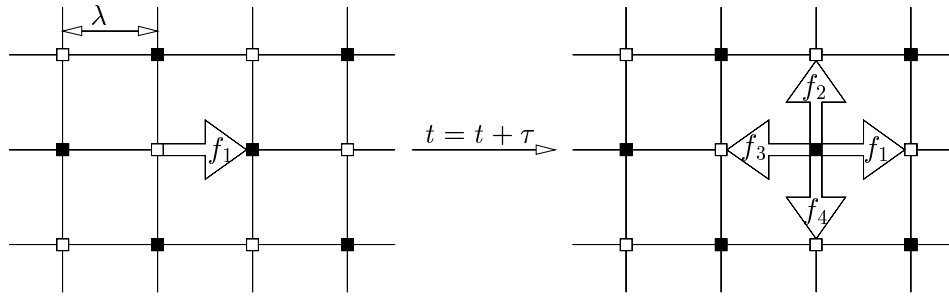


FIGURE 2.2: Illustration of the discrete state and schematic evolution of the automaton in the $4\mathcal{N}2$ lattice.

this freedom for designing our wave automaton. For us, it means (i) to define the macroscopic quantities I and \vec{J} , (ii) to impose conservation laws as given in the equations (2.2) and (2.3) and (iii) impose the additional condition corresponding to (2.1) with the result that the evolution of I is given by a wave equation. So, the two conserved quantities are defined as follows:

$$\text{A scalar: } I \doteq \frac{1}{N+1} \sum_j f_j \quad (2.7)$$

$$\text{A vector: } \vec{J} \doteq \frac{1}{N} \sum_j \vec{v}_j f_j, \quad (2.8)$$

Note that if f is interpreted as the usual distribution function of a multi-particles system, namely

$$\iiint f(\vec{v}) d^3v$$

represents the probability to find a particle having a velocity near \vec{v} , then I is simply the density and \vec{J} the momentum density. In our case this interpretation is not strictly valid because, in general, f_i may take negative values and so may hardly represent an averaged number of particles. That is why we call I and \vec{J} generalized charge and current densities, respectively.

The next step is to write the explicit form of $f^{(0)}$. If we suppose that $f^{(0)}$ is precisely a state of local equilibrium in the sense of the statistical mechanics, it cannot take any form and must depend on f , \vec{r} and t only through the conserved macroscopic variables. Indeed, it is a local interpretation of the first postulate of the statistical mechanics stating that all the micro-states of a system in equilibrium are equiprobable, and a micro-state is completely specified by macroscopic variable like energy, volume, etc. This is precisely the ingredient to make the equation 2.6 a lattice BGK[19, 20] (LBGK) formulation of the model.

To model wave, we choose the following linear definition of $f^{(0)}$:

$$f_i^{(0)} = a_i I + b_i \frac{\vec{v}_i \vec{J}}{v^2}. \quad (2.9)$$

The parameters a_i and b_i must be chosen to guarantee the conservation of I and \vec{J} . Following the definitions (2.7) and (2.8), we must impose

$$\frac{dI}{dt} = \sum_i \frac{df_i}{dt} = 0, \text{ and } \frac{d\vec{J}}{dt} = \sum_i \frac{df_i}{dt} \vec{v}_i = 0 \quad (2.10)$$

where d/dt is the *particular time derivative* referring to the Euler representation of a fluid; in its discretized form, this means

$$\frac{df_i}{dt} \doteq f_i(t + \tau, \vec{r} + \tau \vec{v}_i) \Leftrightarrow f_i(t, \vec{r})$$

With equation (2.6), (2.7) and (2.8), equations 2.10 amount to requiring that

$$\sum_j f_j^{(0)} = (N + 1)I, \quad \text{and} \quad \sum_j v_{j\alpha} f_j^{(0)} = N J_\alpha \quad (2.11)$$

Using expression (2.9) for $f^{(0)}$ we obtain

$$\begin{aligned} I \sum_j a_j + \frac{J_\alpha}{v^2} \sum_j b_j v_{j\alpha} &= (N + 1)I \quad \text{and} \\ I \sum_j a_j v_{j\alpha} + \frac{J_\beta}{v^2} \sum_j b_j v_{j\alpha} v_{j\beta} &= N J_\alpha \end{aligned}$$

To further extract conditions on a_i and b_i , the properties of the lattice has to be specified. We consider general regular \mathcal{N} -dimensional lattices where the properties $\sum_j v_{j\alpha} = 0$ and $\sum_j v_{j\alpha} v_{j\beta} = K \delta_{\alpha\beta}$ hold. To determine the lattice-dependent constant K , the idea is to compute $\sum_j v_{j\alpha} v_{j\beta} v_{k\beta}$. For a given j and k , $v_{j\beta} v_{k\beta}$ is one of the following scalar products (for the case of a square lattice $4\mathcal{N}2$):

$$\begin{aligned} \vec{v}_k \vec{v}_k &= v^2 \\ \vec{v}_{k+1} \vec{v}_k &= 0 \\ \vec{v}_{k+2} \vec{v}_k &= \Leftrightarrow v^2 \\ \vec{v}_{k+3} \vec{v}_k &= 0 \end{aligned}$$

thus

$$\sum_j v_{j\alpha} v_{j\beta} v_{k\beta} = 2v^2 v_{k\alpha} = 2v^2 v_{k\beta} \delta_{\alpha\beta}$$

finally

$$K = 2v^2$$

More generally one finds

$$K = 2v^2 \quad \text{for the } 4\mathcal{N}2 \text{ lattice} \quad (2.12)$$

$$K = \frac{3v^2}{2} \quad \text{for } 3\mathcal{N}2, \text{ and} \quad (2.13)$$

$$K = 3v^2 \quad \text{for } 6\mathcal{N}2 \quad (2.14)$$

If $|\vec{v}_i| = v$ then the directions cannot be distinguished and we must have $a_{i \neq 0} = a$ and $b_{i \neq 0} = b$ and this will be assumed for the remaining of this section. But if the lattice may contain different speeds, as is the case in the Cartesian lattice with diagonals *e.g.* $8\mathcal{N}2$ for which $K = 6v^2$ ($v_{\text{diag}} = v\sqrt{2}$), we may consider different coefficients: $b_i = b_i(|\vec{v}_i|)$. But this case is somewhat special and we shall return later on the little opportuneness of this choice.

After straightforward calculations, the conservation laws imply:

$$a_0 + Na = N + 1 \quad \text{and} \quad b = \frac{Nv^2}{K} \quad (2.15)$$

At this stage of the derivation we have formally given the evolution rule of our microscopic model with equation (2.6). We then have defined macroscopic variables with (2.7) and (2.8). With (2.9) and (2.15), we have built an equilibrium distribution so that it is a linear function of the macroscopic variables which ensure the microscopic conservation of these variables. Thus our model has the ingredient of a discretized wave. The next section will establish that our microscopic conservative dynamics translates in the expected continuity equations (2.2) and (2.3) once the system is considered from a macroscopic (continuous) point of view, and this will eventually show that the model is well suited to simulate a continuous wave.

2.1.1 Continuity equations for I and \vec{J}

We will now calculate, from the microscopic equations governing the behavior of the dynamics of the flux f_i , macroscopic equations, namely equations for I and \vec{J} in the approximation that quantities varies smoothly with space and time. By definition of \vec{v}_i we have $\lambda = |\vec{v}_i|\tau$, where λ is the lattice spacing. In the macroscopic approximation, we can write the discrete particular derivative d/dt as a second order Taylor expansion in τ :

$$\frac{d}{dt} \approx \tau \partial_t + \tau v_{i\alpha} \partial_\alpha + \frac{\tau^2}{2} \partial_t^2 + \frac{\tau^2}{2} \partial_\alpha \partial_\beta v_{i\alpha} v_{i\beta} + \tau^2 \partial_t v_{i\alpha} \partial_\alpha \quad (2.16)$$

The question is to calculate the dynamics of our automata from a macroscopic point of view, namely where the lattice is no longer visible. This is achieved by

supposing “small” the following dimensionless parameter

$$\epsilon = \lambda/L$$

where L represent a macroscopic physical length scale of the problem. By a change of variable, we introduce the macroscopic space variable $r' = \epsilon r$ and consequently $\partial_\alpha = \epsilon \partial'_\alpha$. Thus, using (2.16), equation (2.6) becomes in the new space variable:

$$\left(\tau \partial_t + \epsilon \tau v_{i\alpha} \partial'_\alpha + \frac{\tau^2}{2} \partial_t^2 + \epsilon^2 \frac{\tau^2}{2} \partial'_\alpha \partial'_\beta v_{i\alpha} v_{i\beta} + \epsilon \tau^2 \partial_t v_{i\alpha} \partial'_\alpha \right) f_i = \frac{1}{\xi} (f_i^{(0)} \Leftrightarrow f_i) \quad (2.17)$$

Because of its linearity, equation (2.17) allows a direct solution as we shall see in section 2.2. However, here we will find solutions for f having the form $f = f^{(0)} + \epsilon f^{(1)} + \epsilon^2 f^{(2)} + \dots$. Namely an expansion, or a perturbative analysis around the local equilibrium distribution $f^{(0)}$. The reasons to follow this somewhat more complicated approach, is because it is a standard lattice Boltzmann method used in a simplified context, and a way to straightforward generalization to higher dimension without having to resolve growing matrix. Finding solutions as $\epsilon \rightarrow 0$ is a subtle problem since different time scales may intermix with the result that some terms may diverge as a function of time (so called secular divergence[21]). To overcome these difficulties, we will make use of the multi-scale analysis formalism to ensure a coherent treatment of all the orders. To this end, we associate with equation (2.17) a new equation for a new quantity $F_i(t_1, t_2, r_1)$:

$$\begin{aligned} & \left(\tau (\epsilon \partial_{t_1} + \epsilon^2 \partial_{t_2}) + \epsilon \tau v_{i\alpha} \partial'_\alpha + \frac{\tau^2}{2} (\epsilon \partial_{t_1} + \epsilon^2 \partial_{t_2})^2 + \epsilon^2 \frac{\tau^2}{2} \partial'_\alpha \partial'_\beta v_{i\alpha} v_{i\beta} \right. \\ & \quad \left. + \epsilon \tau^2 (\epsilon \partial_{t_1} + \epsilon^2 \partial_{t_2}) v_{i\alpha} \partial'_\alpha \right) F_i(t_1, t_2, r_1) \\ & = \frac{1}{\xi} (F_i^{(0)}(t_1, t_2, r_1) \Leftrightarrow F_i(t_1, t_2, r_1)) \end{aligned} \quad (2.18)$$

where t_1 and t_2 are new independent time variables. ∂_t has been replaced by $\epsilon \partial_{t_1} + \epsilon^2 \partial_{t_2}$ so it is immediately clear that any solutions of (2.18) considered on the line

$$t_1 = \epsilon t, \quad t_2 = \epsilon^2 t \quad (2.19)$$

becomes a solution of (2.17). The interesting point is that for times outside the line (2.19), the F_i make no sense for our problem and this allows us to choose solutions having the form of a perturbative expansion

$$F_i = F_i^{(0)} + \epsilon F_i^{(1)} + \epsilon^2 F_i^{(2)} + \dots \quad (2.20)$$

with the important property that all terms $F^{(n)}$, $n = 0, 1, 2, \dots$ are well defined and bounded for any time. We can now introduce the ansatz (2.20) in (2.18) and collect the terms with corresponding power of ϵ . All forthcoming considerations are restricted to the time line (2.19) so that further distinction between F and f may be dropped. Now, collecting the terms of order ϵ we have

$$\left(\tau \partial_{t_1} + \tau v_{i\alpha} \partial'_\alpha \right) f_i^{(0)} = \frac{\Leftrightarrow 1}{\xi} f_i^{(1)} \quad (2.21)$$

because $f_i^{(l)}$ are chosen finite for every time and because of the conditions given in (2.11), we obtain after summation over i

$$\sum_i f_i^{(1)} = 0 \quad \Rightarrow \quad \partial_{t_1} I + \frac{N}{N+1} \partial'_\alpha J_\alpha = 0 \quad \text{and} \quad (2.22)$$

$$\sum_i v_{i\gamma} f_i^{(1)} = 0 \quad \Rightarrow \quad \partial_{t_1} J_\alpha + \partial'_\beta S_{\alpha\beta}^{(0)} = 0, \quad (2.23)$$

where

$$S_{\alpha\beta}^{(l)} = \frac{1}{N} \sum_i v_{i\alpha} v_{i\beta} f_i^{(l)}$$

is the stress tensor associated with $f^{(l)}$. The same can be done with the terms of order ϵ^2 and we have:

$$\partial_{t_2} I + \frac{\tau}{2} \partial_{t_1}^2 I + \frac{N}{N+1} \frac{\tau}{2} \partial'_\alpha \partial'_\beta S_{\alpha\beta}^{(0)} + \frac{N}{N+1} \tau \partial'_\alpha \partial_{t_1} J_\alpha = 0 \quad \text{and} \quad (2.24)$$

$$\partial_{t_2} J_\gamma + \frac{\tau}{2} \partial_{t_1}^2 J_\gamma + \partial'_\alpha S_{\alpha\beta}^{(1)} + \frac{\tau}{2} \partial'_\alpha \partial'_\beta T_{\alpha\beta\gamma}^{(0)} + \tau \partial_{t_1} \partial'_\alpha S_{\alpha\gamma}^{(0)} = 0 \quad (2.25)$$

where $NT_{\alpha\beta\gamma}^{(0)} = \sum_i v_{i\alpha} v_{i\beta} v_{i\gamma} f_i^{(0)}$ is a third order tensor. Equation (2.24) and (2.25) can be simplified using (2.22) and (2.23). The next step is to restore the original variables t and r where it is possible and we find the general form of the equations for the macroscopic variables I and \vec{J} . It is independent from the explicit form of the equilibrium distribution $f^{(0)}$ providing it was tuned to microscopically conserve I and \vec{J} . Not surprisingly at all, these continuous equations take the form of two conservation laws:

$$\partial_t I + \frac{N}{N+1} \partial_\alpha J_\alpha = 0 \quad (2.26)$$

$$\partial_t J_\beta + \partial_\alpha \left(S_{\alpha\beta} + \frac{\tau}{2} (\epsilon \partial_{t_1} S_{\alpha\beta}^{(0)} + \partial_\gamma T_{\alpha\beta\gamma}^{(0)}) \right) = 0 \quad (2.27)$$

where $S_{\alpha\beta} = S_{\alpha\beta}^{(0)} + \epsilon S_{\alpha\beta}^{(1)}$. Equation (2.26) is a continuity equation corresponding to (2.2) and equation (2.27) corresponds to (2.3). with a stress tensor, namely the expression for the flux of $J_{beta\alpha}$ in the direction α having a somewhat disagreeable form with three terms; the second of them containing explicit references to the lattice (τ, ϵ). The next section focus on these terms and will establish the conditions under which the last expected relation (2.1) (*i.e.* the constitutive equation) holds for our model; this condition reads

$$\Pi_{\alpha\beta} \doteq S_{\alpha\beta} + \frac{\tau}{2}(\epsilon \partial_{t_1} S_{\alpha\beta}^{(0)} + \partial_\gamma T_{\alpha\beta\gamma}^{(0)}) \propto I \delta_{\alpha\beta} \quad (2.28)$$

2.1.2 The stress tensor $\Pi_{\alpha\beta}$

We must now calculate the stress tensor $\Pi_{\alpha\beta}$ given by (2.28) and show how it can satisfy the condition expressed in (2.1). For this purpose we shall successively evaluate the terms $S_{\alpha\beta}$ and $T_{\alpha\beta\gamma}$. Recalling that $S_{\alpha\beta} \approx S_{\alpha\beta}^{(0)} + \epsilon S_{\alpha\beta}^{(1)}$ we shall firstly calculate the zeroth order of S given by

$$\begin{aligned} S_{\alpha\beta}^{(0)} &= \frac{1}{N} \sum_i v_{i\alpha} v_{i\beta} f_i^{(0)} \\ &= \frac{1}{N} \left(aI \sum_i v_{i\alpha} v_{i\beta} + \frac{N}{K} J_\gamma \sum_i v_{i\alpha} v_{i\beta} v_{i\gamma} \right) \\ &= \frac{K}{N} aI \delta_{\alpha\beta} \end{aligned} \quad (2.29)$$

Before to calculate the first order term $S^{(1)}$ we must solve equation (2.21) in order to find a solution for $\mathbf{f}^{(1)}$:

$$f_i^{(1)} = \Leftrightarrow \xi \tau \left(\partial_{t_1} f_i^{(0)} + \partial'_\alpha v_{i\alpha} f_i^{(0)} \right)$$

$f^{(0)}$ depends on t_1 only through I and J , thus

$$f_i^{(1)} = \Leftrightarrow \xi \tau \left(\frac{\partial f^{(0)}}{\partial I} \partial_{t_1} I + \frac{\partial f_i^{(0)}}{\partial J_\alpha} \partial_{t_1} J_\alpha + v_{i\alpha} \partial'_\alpha \left(a_i I + \frac{b_i}{v^2} v_{i\beta} J_\beta \right) \right)$$

with (2.9)

$$f_i^{(1)} = \Leftrightarrow \xi \tau \left(a_i \partial_{t_1} I + \frac{b_i}{v^2} v_{i\alpha} \partial_{t_1} J_\alpha + a v_{i\alpha} \partial'_\alpha I + \frac{b_i}{v^2} v_{i\alpha} v_{i\beta} \partial'_\alpha J_\beta \right)$$

from the conservation laws (2.22) and (2.23) we have

$$f_i^{(1)} = \xi \tau \left(\frac{N}{N+1} a_i \partial'_\alpha J_\alpha + \frac{b_i}{v^2} v_{i\alpha} \partial'_\beta S_{\alpha\beta}^{(0)} \Leftrightarrow a v_{i\alpha} \partial'_\alpha I \Leftrightarrow \frac{b_i}{v^2} v_{i\alpha} v_{i\beta} \partial'_\alpha J_\beta \right)$$

with (2.29)

$$f_i^{(1)} = \xi\tau \left(\frac{N}{N+1} a_i \partial'_\alpha J_\alpha + \frac{b_i K}{v^2 N} a v_{i\alpha} \partial'_\alpha I \Leftrightarrow a v_{i\alpha} \partial'_\alpha I \Leftrightarrow \frac{b_i}{v^2} v_{i\alpha} v_{i\beta} \partial'_\alpha J_\beta \right)$$

with (2.15), finally

$$f_i^{(1)} = \xi\tau \left(\frac{N}{N+1} a_i \delta_{\alpha\beta} \Leftrightarrow \frac{N}{K} v_{i\alpha} v_{i\beta} \right) \partial'_\alpha J_\beta \quad (2.30)$$

The first order of S may be now calculated

$$\begin{aligned} S_{\alpha\beta}^{(1)} &= \frac{1}{N} \sum_i v_{i\alpha} v_{i\beta} f_i^{(1)} \\ &= \frac{\xi\tau}{N} \left(\frac{KN}{N+1} a \partial'_\gamma J_\gamma \delta_{\alpha\beta} \Leftrightarrow \frac{N}{K} \partial'_\gamma J_\delta E_{\alpha\beta\gamma\delta} \right) \end{aligned} \quad (2.31)$$

where $E_{\alpha\beta\gamma\delta} = \sum_i v_{i\alpha} v_{i\beta} v_{i\gamma} v_{i\delta}$ is a fourth order tensor which turns out to be anisotropic on square lattices. This a well known “strong” anisotropy that does not disappear in the limit $\lambda \rightarrow 0$. Indeed, a straightforward calculation shows that if $E_{\alpha\beta\gamma\delta}$ is calculated for a particular orientation of the lattice directions with respect to the coordinates system, we obtain a fourth order tensor which is explicitly function of the orientation. This is of course not desirable but we shall see below how this problem is solved.

Now that we have calculated $S_{\alpha\beta}$, it remains $T_{\alpha\beta\gamma}^{(0)}$ to get an expression for $\Pi_{\alpha\beta}$ out of equation (2.28):

$$\begin{aligned} T_{\alpha\beta\gamma}^{(0)} &= \frac{1}{N} \sum_i v_{i\alpha} v_{i\beta} v_{i\gamma} f_i^{(0)} \\ &= \frac{1}{K} J_\delta E_{\alpha\beta\gamma\delta} \end{aligned} \quad (2.32)$$

Now collecting the terms in (2.29), (2.30), (2.31), (2.32) and using again (2.22) and the change of variable $\partial_\alpha = \epsilon \partial'_\alpha$, we can write $\Pi_{\alpha\beta}$:

$$\Pi_{\alpha\beta} = a \frac{K}{N} I \delta_{\alpha\beta} + \left(\xi \Leftrightarrow \frac{1}{2} \right) \left(a \frac{K}{N+1} \tau \partial_\gamma J_\gamma \delta_{\alpha\beta} \Leftrightarrow \frac{1}{K} \tau \partial_\gamma J_\delta E_{\alpha\beta\gamma\delta} \right) \quad (2.33)$$

To satisfy (2.1) up to a constant factor and thus make I be governed by a wave equation we must cancel the second term in (2.33). We see that this is simply achieved by setting our free relaxation time parameter $\xi = 1/2$. The nature of the cancelled terms is of diffusive type: an isotropic contribution expressed as a divergence of \vec{J} and an anisotropic lattice (for the square lattice) contribution due to the fourth order tensor $E_{\alpha\beta\gamma\delta}$. Clearly, these terms are breaking the time

reversal invariance. This is undesirable in the context of wave propagation that's why we shall make use of the still free parameter ξ to cancel these terms. But if we think of simulating amortized waves, we may have some interest in keeping diffusive contribution in our model. The choice $\xi = 1/2$ corresponds precisely to the 0-viscosity case for the complete fluid model. For the fluid LBGK model the 0-viscosity limit turns out to be numerically instable but because of the linearity of our $f_i^{(0)}$'s this is fortunately not the case for the wave model.

2.1.3 The wave equation

The results derived in the previous section yield the following three first order equations for macroscopic time and length scales:

$$\Pi_{\alpha\beta} = a \frac{K}{N} I \delta_{\alpha\beta} \quad (2.34)$$

$$\partial_t I + \frac{N}{N+1} \partial_\alpha J_\alpha = 0 \quad (2.35)$$

$$\partial_t J_\beta + \partial_\alpha \Pi_{\alpha\beta} = 0 \quad (2.36)$$

where I and \vec{J} are defined by (2.7) and (2.8) respectively, providing that the following conditions are added to the formulation of LBGK model given in (2.6):

$$f_i^{(0)} = a_i I + b_i \frac{\vec{v}_i \vec{J}}{v^2}, \quad (2.37)$$

$$a_0 + Na = N + 1 \quad \text{and} \quad b = \frac{Nv^2}{K}, \quad (2.38)$$

$$\xi = \frac{1}{2} \quad (2.39)$$

Diffusive contributions or any other corrections are of an order of magnitude smaller than λ^2/τ . This means that our model shows, at a macroscopic scale, wave propagation for I , namely

$$\partial_t^2 I \Leftrightarrow a \frac{K}{N+1} \nabla^2 I = 0, \quad 0 < a < \frac{N+1}{N}$$

with a propagation speed

$$c = \sqrt{a \frac{K}{N+1}}$$

that is defined at the microscopic level of our automaton. As a matter of fact, $a = a(\vec{r})$ enters the definition of the local equilibrium distribution $f^{(0)}$ (see (2.9)) independently at each site of lattice.

Clearly we must have $a > 0$ to have a proper wave equation. The upper limit for a corresponding to $a_0 = 0$ will be easily interpreted with the particle paradigm. but

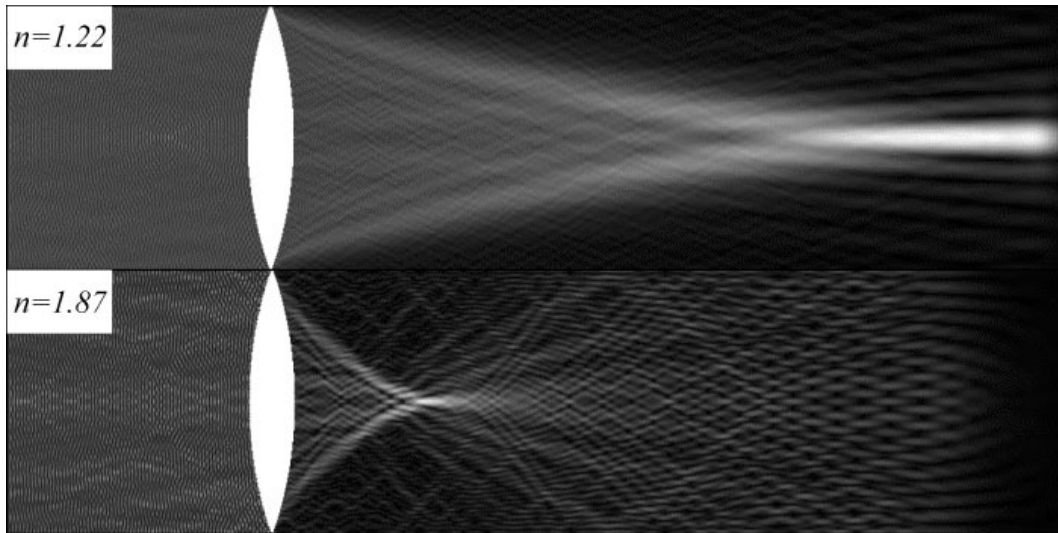


FIGURE 2.3: *Lattice Boltzmann wave model and focusing of plane waves incident on a region with different refraction index $n = 1.22$ and $n = 1.87$: simulation of convex lenses (shown in white on the figure). The source consists of a coherent, synchronous oscillation of all the left most sites of the domain. The grey levels indicates from black to white the intensity of wave at each site.*

we shall see later on that this upper limit is also a consequence from an additional condition imposed on the model to ensure numerical stability: unitarity.

The units are correct since $K \propto v^2$ and the other coefficients are dimensionless. The possibility to adjust the propagation speed on every site is a very interesting property of our model; so we can define a local index of refraction $n = c_0/c$, where the maximum wave speed c_0 (the speed in vacuum if reference is made to light waves) is determined by the lattice nature. It corresponds to the case where $a_0 = 0$, namely the local equilibrium distribution no longer depends on the rest flux f_0 . In other words and as expected, eliminating rest “particles” maximize the propagation speed and we have

$$a_0 = 0 \quad \Rightarrow \quad c_0 = \sqrt{\frac{K}{N}}, \quad n = \sqrt{\frac{N+1}{aN}} \quad (2.40)$$

Figure 2.3 illustrates the propagation of a plane wave through a lens shaped medium. The lattice sites outside the lens have a collision term Ω corresponding to the case $a_0 = 0$ while the internal site have $a_0 > 0$, namely $n > 1$. Due to the change of speed between the two media, we observe a focusing of the incident plane wave.

Following (2.12)-(2.14), all lattices such that $|v_i| = v$ we have $c_0 = v/\sqrt{2}$. This means that the wave propagation speed is always lower than the speed at which the information contained in the flux fare traveling. In conclusion, the derivation of our model has clarified a sufficient set of basic physical ingredients hidden

behind the wave phenomenon: conservation of a scalar plus conservation of its current defined by the continuity equation (2.38) plus linearity (2.37) plus time reversal invariance (2.39) of the process. Indeed, we will show in the next section how $\xi = 1/2$ is related to time reversal invariance.

2.1.4 The time reversal invariance, viscosity

In the previous section we derived a linear lattice Boltzmann-like model to simulate wave propagation of the scalar I on a square lattice. Because of its linearity, (2.6) may be rewritten by mean of a propagation matrix W as follows:

$$f'_i = \left(1 \Leftrightarrow \frac{1}{\xi}\right) f_i + \frac{1}{\xi} f_i^{(0)} \doteq \sum_{j=0}^4 W_{ij} f_j \quad (2.41)$$

f' designates the outgoing flux, namely the flux after the collision but before the free displacement. We saw that the relaxation time ξ should be set to $1/2$ in order to get a good wave model. This amounts to impose time reversal invariance in our model. Time reversal invariance is a fundamental symmetry of microscopic physics and is clearly satisfied by the wave equation which has only a second time derivative. The dynamics is reversible in time if, after reversing the direction of the flux, the system traces back its own past. In our case, reversing the flux is achieved with a reversal matrix R given by

$$R = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{pmatrix} \quad (2.42)$$

So that time reversal invariance for our dynamics means

$$WRf' = Rf$$

using the definition (2.9) of $f^{(0)}$ we obtain

$$(WRf')_i = \left(1 \Leftrightarrow \frac{1}{\xi}\right) (Rf')_i + \frac{1}{\xi} \left(a_i I(Rf') + b_i \frac{\vec{v}_i}{v^2} \vec{J}(Rf') \right)$$

Since we have $I(Rf') = I(f)$ and $\vec{J}(Rf') = \Leftrightarrow \vec{J}(f)$:

$$\begin{aligned} &= \left(1 \Leftrightarrow \frac{1}{\xi}\right) f'_{i+2} + \frac{1}{\xi} \left(a_i I(f) \Leftrightarrow b_i \frac{\vec{v}_i}{v^2} \vec{J}(f) \right) \\ &= \left(1 \Leftrightarrow \frac{1}{\xi}\right) \left(\left(1 \Leftrightarrow \frac{1}{\xi}\right) f_{i+2} + \frac{1}{\xi} f_{i+2}^{(0)} \right) + \frac{1}{\xi} f_{i+2}^{(0)} \\ &= \left(1 \Leftrightarrow \frac{1}{\xi}\right)^2 f_{i+2} + \frac{1}{\xi} \left(2 \Leftrightarrow \frac{1}{\xi} \right) f_{i+2}^{(0)} \end{aligned}$$

This means that

$$\left(1 \Leftrightarrow \frac{1}{\xi}\right)^2 = 1 \quad \text{and} \quad \frac{1}{\xi} \left(2 \Leftrightarrow \frac{1}{\xi}\right) = 0$$

whose solution is $\xi = 1/2$. In the previous section we saw that $\xi = 1/2$ makes the viscosity or dissipative terms vanish. Indeed, breaking the time reversal symmetry is the standard definition of dissipation. We have also seen that the anisotropy due to the square lattice (the fourth-order tensors $E_{\alpha\beta\gamma\delta}$) is also removed by the choice $\xi = 1/2$.

2.1.5 Dissipation

Pure waves - such as electromagnetic waves in vacuum - propagate without dissipation. This means that the phenomena is time reversible. Introducing dissipation in our model or breaking the time reversal invariance of its dynamics may be important for different reasons. On the one hand, real media are never ideal and always dissipate some energy during time. On the other hand, our simulations will always take place in finite domains and we shall see later, that we must gradually absorb the waves on the boundary if we need to simulate a far larger space.

In our model, several ways can be proposed to include dissipation. We can dissipate the waves without dissipating energy, or we can dissipate the waves by removing “particles”. A third way to dissipate wave is equivalent to a statistical mixture of free propagation and perfect reflection. The next four subsections present the alternatives and define what is meant by perfect reflection.

Dissipation I

If we want to introduce properly dissipation in our model, it is not enough to chose $\xi > 1/2$, we have to work with lattices whose tensor $E_{\alpha\beta\gamma\delta}$ is symmetric. This is not the case for the square lattice $4\mathcal{N}2$, since in particular orientation of the coordinate system, the four possible directions are

$$\begin{aligned} \vec{v}_1 &= v \left(\cos(\phi), \sin(\phi) \right), \\ \vec{v}_2 &= v \left(\Leftrightarrow \sin(\phi), \cos(\phi) \right), \\ \vec{v}_3 &= v \left(\Leftrightarrow \cos(\phi), \Leftrightarrow \sin(\phi) \right), \\ \vec{v}_4 &= v \left(\sin(\phi), \Leftrightarrow \cos(\phi) \right) \end{aligned}$$

where v is the lattice spacing divided by the time step. As the square lattice is invariant under the inversion of the space coordinates x and y , $E_{\alpha\beta\gamma\delta}$ can only

have the following forms established by a direct calculation:

$$\begin{aligned}\sum_{i=1}^4 (\vec{v}_i)_x^4 &= v^4 \left(\frac{3}{2} + \frac{1}{2} \cos(4\phi) \right), \\ \sum_{i=1}^4 (\vec{v}_i)_x^3 (\vec{v}_i)_y &= v^4 \frac{1}{2} \sin(4\phi), \\ \sum_{i=1}^4 (\vec{v}_i)_x^2 (\vec{v}_i)_y^2 &= v^4 \left(\frac{1}{2} \Leftrightarrow \frac{1}{2} \cos(4\phi) \right)\end{aligned}$$

The anisotropy appears clearly as a general dependence in ϕ . It is known from the general cellular automata theory [22] that $E_{\alpha\beta\gamma\delta}$ is isotropic on hexagonal lattices $6\mathcal{N}2$ and have the following characteristic form:

$$E_{\alpha\beta\gamma\delta} = v^4 \frac{N}{8} (\delta_{\alpha\beta}\delta_{\gamma\delta} + \delta_{\alpha\gamma}\delta_{\beta\delta} + \delta_{\alpha\delta}\delta_{\beta\gamma})$$

This allows us to rewrite (2.33) as follows

$$\Pi_{\alpha\beta} = a \frac{K}{N} I \delta_{\alpha\beta} + \tau \left(\xi \Leftrightarrow \frac{1}{2} \right) \left(a \frac{K}{N+1} \partial_\gamma J_\gamma \delta_{\alpha\beta} \Leftrightarrow v^4 \frac{N}{8K} (\partial_\gamma J_\gamma \delta_{\alpha\beta} + \partial_\alpha J_\beta + \partial_\beta J_\alpha) \right)$$

and we obtain together with (2.35) and (2.36) $N = 6$, $K = 3v^2$ the amortized wave equation for I :

$$\boxed{\partial_t^2 I \Leftrightarrow c^2 \nabla^2 I \Leftrightarrow D \nabla^2 \partial_t I = 0}$$

where

$$\begin{aligned}c^2 &= \frac{3}{7} a v^2, \\ D &= \tau \left(\xi \Leftrightarrow \frac{1}{2} \right) \left(\frac{v^2}{4} \Leftrightarrow c^2 \right), \\ \text{and } 0 &< a < \frac{7}{6}\end{aligned}$$

In most computer, the data structure representing a hexagonal lattice is more difficult to work with than a regular square lattice which is naturally implemented as an array of data. There is a possible alternative also developed in the context of fluid models to work on square lattice connecting the nearest and second nearest (in diagonal) neighbors. This case is equivalent to the superposition of two square lattices: the first one (odd) with $|\vec{v}_i| = v$ and the second (even), rotated by $\pi/4$ having $|\vec{v}_i| = v\sqrt{2}$; see figure 2.1, the case $8\mathcal{N}2$. We can observe that orientation dependence in ϕ may be canceled if the contribution of the even velocities (coming the factor $4v^4$) are counted $1/4$ of the contribution of the odd velocities.

This means that an isotropic dissipative wave model is, in principle, possible on the square lattice with 8 velocities providing that a different b is chosen on each

sub-lattice $4\mathcal{N}2$: $b_{\text{odd}}/b_{\text{even}} = 4$. But it turns out that bad propagation pattern is observed on $8\mathcal{N}2$, especially in the limit $\xi \rightarrow 1/2$ where kinds of instabilities appears. This instabilities are striking since they remind the instabilities appearing in the fluid flows model [23] in the low viscosity limit but they were not expected in our fully linear model. On the other hand, having different magnitudes for the propagation of informations is spurious in the framework of Huygens' principle because a certain "distance" beyond locality is allowed for the action.

This failure to have a correct stable model on $8\mathcal{N}2$ indicates that one cannot have an arbitrary large propagation speed c by using different links lengths v_i in our lattice model. More precisely, we shall show later, that different v_i prevent the symetrization of the propagation matrix and thus the conservation of a certain class of quadratic forms in \mathbf{f} . The upper limit c_0 depends on the dimensionality \mathcal{N} of the space but is independent from the Bravais lattice chosen. Indeed for a set of N vectors uniformly distributed around the unit circle in a \mathcal{N} -dimensional space, it possible to check that we have $K = N/\mathcal{N}$ and consequently

$$c_0^2 = \frac{1}{\mathcal{N}}$$

In conclusion we have shown how to model dissipative LB waves on the hexagonal lattice $6\mathcal{N}2$. The model allows one to tune both the propagation speed and the dissipation at a microscopic level. The attempt to built an isotropic dissipative wave model on $8\mathcal{N}2$ failed, however we will see next on how to deal with dissipative waves on the simple square lattice $4\mathcal{N}2$.

Dissipation II

The dissipation presented above is a pure consequence of the breaking of time reversal invariance. It is essential to note that both I and \vec{J} remain conserved for any value of ξ , dissipative or not. In this circumstance we can see that a definition of a local energy of our system is not straightforwardly given by a function of I and \vec{J} . Indeed, we shall see in the next section that an additional quantity may be defined as a (quadratic) function of the f_i 's which behaves like an energy: stays positive, and is conserved if and only if $\xi = 1/2$.

Dissipation may be introduced more crudely on any kind of lattices if one actually breaks the conservation laws for I and \vec{J} . This is a more intuitive approach of dissipation which corresponds to simply multiply the propagation matrix W defined in (2.41) by an absorption factor μ . It is equivalent to removing particles during the collision process. We start with two equations expressing the same dissipation for I and \vec{J} :

$$\begin{aligned} \frac{dI}{dt} &= \Leftrightarrow \alpha I \\ \frac{d\vec{J}}{dt} &= \Leftrightarrow \alpha \vec{J} \end{aligned}$$

If $\xi = 1/2$ these two equations impose new conditions on a_i and b_i which may be summarized by a multiplicative factor in front of W :

$$\mu = 1 \Leftrightarrow \alpha/2. \quad (2.43)$$

This kind of dissipation is easier to deal with and was chosen for the application presented in chapter 4.

Dissipation III

Absorbing media can also be simulated if we multiply a_i entering the definition of $\mathbf{f}^{(0)}$ (2.9) with an attenuation factor $0 < \mu < 1$. This corresponds to hinder the conservation of I (ensured by a careful choice of the a_i 's; see (2.15)) whereas \vec{J} remains conserved (because of the conditions imposed on the b_i 's; see (2.15)). Consequently, this defines a new equilibrium $\tilde{\mathbf{f}}^{(0)}$ such that $\tilde{f}_0^{(0)} = \mu a_0 I$ and $\tilde{f}_i^{(0)} = \mu a_i I + \mathcal{N} \vec{v}_i \vec{J}$ where \mathcal{N} is the dimensionality of the space (We have used the fact that $K = N/\mathcal{N}$ if $|\vec{v}_i| = v$) and we may rewrite (2.41) as follows ($\xi = 1/2$)

$$\begin{aligned} f_i' &= 2\tilde{f}_i^{(0)} \Leftrightarrow f_i \\ f_i' &= 2\mu f_i^{(0)} \Leftrightarrow f_i + (1 \Leftrightarrow \mu) \mathcal{N} \vec{v}_i \vec{J} \\ f_i' &= \mu (2f_i^{(0)} \Leftrightarrow f_i) + (1 \Leftrightarrow \mu) (\mathcal{N} \vec{v}_i \vec{J} \Leftrightarrow f_i) \\ \mathbf{f}' &= \mu W \mathbf{f} \Leftrightarrow (1 \Leftrightarrow \mu) R \mathbf{f} \end{aligned} \quad (2.44)$$

where R is defined by equation (2.42). From (2.44) we see that dissipation or absorption is equivalent to a statistical mixture of free propagation W with probability μ and perfect reflection R with probability $1 \Leftrightarrow \mu$.

Reflection

As mentioned just above, the matrix R introduced in (2.42) for algebraic reasons, may be used to locally define perfect reflecting sites. These reflector-sites are important to simulate conducting boundary conditions. In that case, the evolution of the f_i 's simply becomes

$$f_i(\vec{r}_i + \tau \vec{v}_i, t + \tau) = \sum_{j=0}^4 R_{ij} f_j \quad (2.45)$$

Figure 2.4 shows the focusing effect of a plane wave traveling from a free propagation medium towards a parabolic mirror. The parabolic mirror is a discrete arrangement of sites colliding the incoming f_i 's with R instead of W . The focusing effect emerges as a collective behavior of all these reflectors.

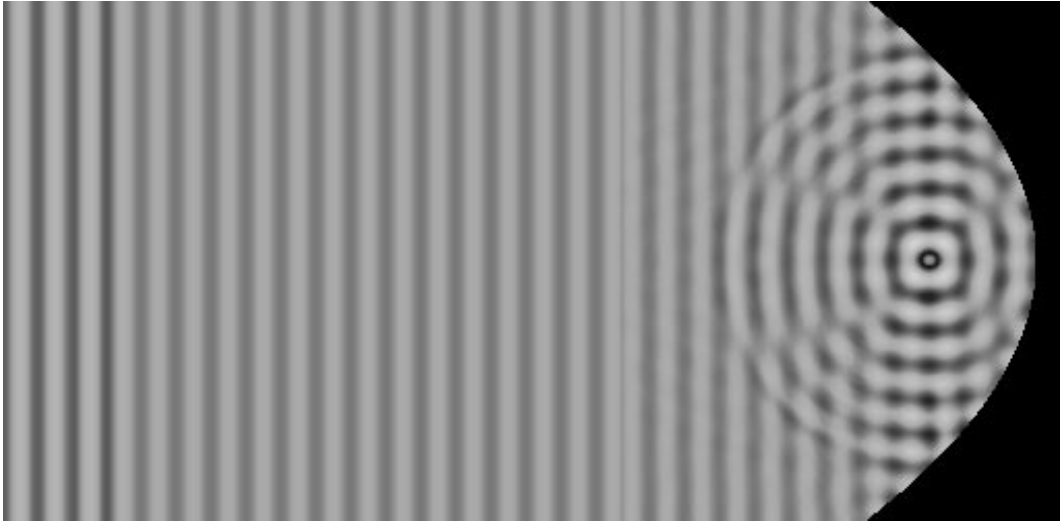


FIGURE 2.4: *Focusing of a plane wave incoming from the left side of the domain and traveling towards an parabolic-shaped arrangement of perfect reflecting sites.*

2.2 Transmission Line Matrix

We have already mentioned in (2.41) that equation (2.6) is linear and thus may be rewritten by mean of a propagation matrix W :

$$f_i(\vec{r}_i + \tau\vec{v}_i, t + \tau) = \sum_{j=0}^4 W_{ij} f_j \quad (2.46)$$

2.2.1 The symmetric 5×5 matrix formulation

For the square lattice $4\mathcal{N}2$ with f_1, f_2, f_3, f_4 and a rest flux f_0 , W_{ij} may be easily calculated from the definitions (2.7) and (2.8) and the conditions (2.37), (2.38) and (2.39). With the connectivity number N , we obtain a matrix which may be easily generalized to the 3-dimensional square lattice $6\mathcal{N}3$. Here below we show the matrix for the 2-dimensional Cartesian case (*i.e.* $N = 4$ and $\mathcal{N} = 2$):

$$W = \frac{2a}{N+1} \cdot \quad (2.47)$$

$$\begin{pmatrix} \frac{N+1}{2a} - N & \frac{N+1}{a} - N & \frac{N+1}{a} - N & \frac{N+1}{a} - N & \frac{N+1}{a} - N \\ 1 & 1 - \frac{N+1}{a}(\frac{1}{2} - \frac{\mathcal{N}}{N}) & 1 & 1 - \frac{N+1}{a}(\frac{1}{2} - \frac{\mathcal{N}}{N}) & 1 \\ 1 & 1 & 1 - \frac{N+1}{a}(\frac{1}{2} - \frac{\mathcal{N}}{N}) & 1 & 1 - \frac{N+1}{a}(\frac{1}{2} - \frac{\mathcal{N}}{N}) \\ 1 & 1 - \frac{N+1}{a}(\frac{1}{2} - \frac{\mathcal{N}}{N}) & 1 & 1 - \frac{N+1}{a}(\frac{1}{2} - \frac{\mathcal{N}}{N}) & 1 \\ 1 & 1 & 1 - \frac{N+1}{a}(\frac{1}{2} - \frac{\mathcal{N}}{N}) & 1 & 1 - \frac{N+1}{a}(\frac{1}{2} - \frac{\mathcal{N}}{N}) \end{pmatrix}$$

This matrix may well be considered as an example for the construction of a general propagation matrix in any dimension \mathcal{N} with any coordination number

N , providing that \vec{v}_i is the opposite of $\vec{v}_{1+i+N/2}$. The scalar factor and the first line remain the same. Elsewhere, matrix elements different from 1 but equal to those given (2.47) are found in each line $i > 1$ at positions $j = i$ (diagonal elements) and $j = 1 + i + N/2$. Let us now have a closer look to the property of this matrix. It is easy to check that the dynamics obtained with this propagation matrix is, by construction, invariant under the space inversion; it means:

$$\begin{aligned} \mathbf{f}' = W\mathbf{f} & \xleftrightarrow[\text{space}]{\text{inverse}} R\mathbf{f}' = WR\mathbf{f} \\ \Leftrightarrow & R^2\mathbf{f}' = RWR\mathbf{f} \\ \Leftrightarrow & W = RWR \end{aligned} \quad (2.48)$$

where R is the reversal matrix defined by (2.42), thus $R^2 = 1$. We have also established in the previous subsection that this dynamics is time reversal invariant since it corresponds to the case $\xi = 1/2$; it means:

$$\begin{aligned} \mathbf{f}' = W\mathbf{f} & \xleftrightarrow[\text{time}]{\text{inverse}} WR\mathbf{f}' = R\mathbf{f} \\ \Leftrightarrow & WRW\mathbf{f} = R\mathbf{f} \\ \Leftrightarrow & WRW = R \end{aligned} \quad (2.49)$$

The symmetry conditions (2.48) and (2.49) implies $W^2 = 1$. The condition $W^2 = 1$ does not mean that $|\mathbf{f}|^2$ (the norm of \mathbf{f} based upon the usual scalar product) is conserved. As a matter of fact it is easy to check that it not the case for W given by (2.47): the sum of the square of the matrix elements in each column is not equal 1 in the general case. However, it is interesting to note that our dynamics conserves U a quadratic form in addition to the quantities I and \vec{J} . In other words, we shall show that there is simple a change of variable that makes W a unitary matrix. Because we already know that $W^2 = 1$ the change of variable we are looking for is such that it symmetries the matrix W . So it is useful to show that it exists a non-singular matrix M_0 such that

$$W' \doteq M_0^{-1}WM_0 \Rightarrow W' = W'^{\perp} \quad (2.50)$$

This is proven here by inspection; one can easily check that the following M_0

$$M_0 = \begin{pmatrix} m_0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} \quad (2.51)$$

where

$$m_0 = \sqrt{\frac{N+1}{a}} \Leftrightarrow N$$

actually symmetrizes the propagation matrix. Then

$$W'^2 = M_0^{-1} W M_0 M_0^{-1} W M_0 = M_0^{-1} W^2 M_0 = 1$$

Now we want to show that $U = |M_0^{-1} \mathbf{f}|^2$ is conserved during the evolution given W applied on \mathbf{f} . We have

$$U \doteq \mathbf{f}^\perp M_0^{-1\perp} M_0^{-1} \mathbf{f}$$

during the collision process $\mathbf{f} \rightarrow W \mathbf{f}$ and U becomes

$$U \rightarrow U' = \mathbf{f}^\perp W^\perp M_0^{-1\perp} M_0^{-1} W \mathbf{f}$$

since $M_0^{-1} W = W' M_0^{-1}$ we obtain

$$U' = \mathbf{f}^\perp M_0^{-1\perp} W'^\perp W' M_0^{-1} \mathbf{f}$$

but W' is symmetric by construction

$$\begin{aligned} U' &= \mathbf{f}^\perp M_0^{-1\perp} W'^2 M_0^{-1} \mathbf{f} \\ U' &= \mathbf{f}^\perp M_0^{-1\perp} M_0^{-1} \mathbf{f} = U \quad \square \end{aligned}$$

In conclusion it is possible to make a change of coordinates $\mathbf{g} = M_0 \mathbf{f}$ so that $\mathbf{g}^\perp \mathbf{g}$ is conserved besides the linear quantities I and \vec{J} . This change of coordinates has no dramatic consequences on our model since it is equivalent to the redefinition of I as

$$I = m_0 g_0 + \sum_{i=1}^N g_i$$

where m_0 has become the “mass” of the rest particles mass of rest particle (thus the notations adopted is justified) whereas the moving particles have a mass $m_i = 1$. Clearly, the mass m_0 may be defined as a real valued quantity for any a as long as

$$a < \frac{N+1}{N}$$

This is the other way to get the upper limit possible for a mentioned earlier in the text. The extreme case $m_0 = 0$ - no rest particles - is equivalent to the case $a_0 = 0$ (before the change of variable) of (2.40) and corresponds to the maximal wave propagation speed allowed on the lattice. We now have a deeper interpretation of this limit: beyond this limit there is no way to symmetrize the propagation matrix W with a diagonal change of variable (a diagonal change of variable is required because, together with W , there is a spreading phase of the f_i 's; we shall return on this point later on), namely W is not equivalent to an unitary matrix and this may finally entail instabilities in the numerical scheme.

For $m_0 = 0$ the rest flux turns out to have an independent evolution (in the original coordinates system): $f_0(t + \tau) = \Leftrightarrow f_0(t)$. Consequently we may drop f_0 and we are left with the following 4×4 , symmetric, propagation matrix:

$$W_{\text{TLM}} = \frac{1}{2} \begin{pmatrix} 1 & 1 & \Leftrightarrow 1 & 1 \\ 1 & 1 & 1 & \Leftrightarrow 1 \\ \Leftrightarrow 1 & 1 & 1 & 1 \\ 1 & \Leftrightarrow 1 & 1 & 1 \end{pmatrix} \quad (2.52)$$

This so-called Transmission Line Matrix (TLM) (2.52) has been proposed and reinvented by several authors for different applications; see [24] for a review in the electromagnetics community. In the electrical engineer community it used to represent the evolution of actual circuits. It is interesting to note that the TLM matrix is issued from a simulation tradition which goes back (long) before the advent of digital computers. This is actually highlighting the profound difference between the numerical simulation and the numerical computation approach. Indeed, the similarity between the behavior of electromagnetic fields and of voltages and currents in electrical networks was extensively used during the first half of the twentieth century, to simulate field problems [25] lying beyond the analytical approach. More recent work to this wave propagation model can be found for instance in [26, 27].

2.2.2 The dispersion relation

Equation like (2.46) may be solved directly to obtain the dispersion relation of the LB wave model. Consider plane-waves solutions

$$f_i(\vec{r}, t) = \exp\left(i(\vec{k}\vec{r} \Leftrightarrow \omega t)\right)$$

of the equation

$$f_i(\vec{r}_i + \tau\vec{v}_i, t + \tau) = \sum_{j=0}^4 W_{ij} f_j$$

For simplicity, consider the case $W = W_{\text{TLM}}$ without rest flux f_0 . After substitution, it is found that the plane wave expression is a solution of the evolution provided that $\exp(i\omega\tau)$ is a solution of the polynome

$$X^4 \Leftrightarrow \kappa X^3 + \kappa X \Leftrightarrow 1 = 0$$

where $\kappa = \cos(\lambda k_1) + \cos(\lambda k_2)$. The quantities $k_{[12]}$ are the spatial components of \vec{k} and $\lambda = v\tau$. Two solutions are not propagation solutions, namely

$$\exp(i\omega\tau) = \pm 1;$$

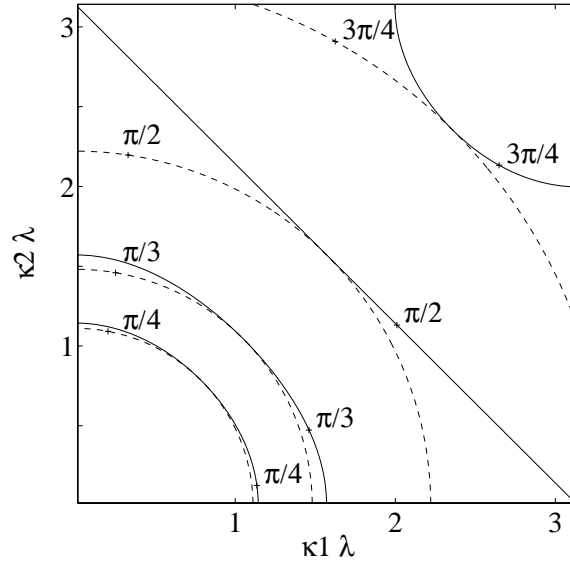


FIGURE 2.5: Contour line of the real part of dispersion relation (2.53) where the value $\omega\tau$ is shown as a function of $k_{[12]}$. The solid line corresponds to the LB wave model and the dashed line to the wave equation. The limiting case for large k 's, namely $\omega\tau = \pi/2$ appears clearly.

they correspond to stationary and flipping spatial distribution of the f 's respectively. These solutions are the so-called staggered invariants, they are purely artifacts of the discretized model. The other solutions are

$$\exp(i\omega\tau) = \frac{1}{2}(\kappa \pm i\sqrt{4 \Leftrightarrow \kappa^2}) \quad (2.53)$$

The behavior of the dispersion relation is shown on figure 2.5. We may first observe that the LB wave model is exact for diagonal propagation where $k_1 = k_2$. The discrete Fourier transform theory tells us that the possible k on a lattice are given by

$$k = \frac{2\pi}{\lambda} \frac{i}{N} \quad \text{where } i \in \{\Leftrightarrow \frac{N}{2}, \dots, 0, 1, \dots, \frac{N}{2} \Leftrightarrow 1\}$$

where N is the (even) number of site along the considered dimension. The limiting case $k\lambda = \pi$ is clearly visible on figure 2.5; it corresponds to $\omega\tau = \pi/2$. In other words the oscillation period $T = 2\pi/\omega$ must be larger than 4τ . This limit is easy to understand: if you want to describe properly an oscillation of say, a pendulum in discrete time, you will need at least four time steps corresponding to four situations where the “speed” of the pendulum is zero and maximal. One may argue that only two time steps corresponding to the situation where the speed is zero (extrema of the oscillation) may suffice but this case actually correspond to the “flipping” situation, or staggered invariant mentioned above.

A Taylor expansion of (2.53) gives, with $k^2 = k_1^2 + k_2^2$

$$i\omega\tau \Leftrightarrow \frac{\omega^2\tau^2}{2} + O(\tau^3) = \Leftrightarrow \frac{\lambda^2 k^2}{4} \pm i \frac{\lambda k}{\sqrt{2}} + O(\lambda^3) \quad (2.54)$$

Taking the continuous limit of this sort of equation always raises the question to know how λ and τ , the two parameters of the discretisation, are related to each other when they tend to zero. If we assume that both convective and diffusive phenomena are present, τ is expected to have components of the order λ and also λ^2 . We shall apply here the multi-scale formalism introduced in section 2.1.1, namely associate with equation (2.54) a new equation for the macroscopic independent variable τ_1 , τ_2 and λ , with ϵ the parameter which goes to zero as the continuous limit is taken:

$$i\omega\epsilon\tau_1 + i\omega\epsilon^2\tau_2 \Leftrightarrow \frac{\omega^2\epsilon^2\tau_1^2}{2} + O(\epsilon^3) = \Leftrightarrow \frac{\epsilon^2\lambda_1^2 k^2}{4} \pm i \frac{\epsilon\lambda_1 k}{\sqrt{2}} + O(\epsilon^3) \quad (2.55)$$

every solution of (2.55) considered for

$$\begin{aligned} \epsilon\tau_1 + \epsilon^2\tau_2 &= \tau \\ \epsilon\lambda_1 &= \lambda \end{aligned}$$

is a solution of (2.54). We can now collect the various terms of equation (2.55) according to their order of magnitude. We obtain

$$O(\epsilon) : \quad \omega = \pm \frac{1}{\sqrt{2}} \frac{\lambda_1}{\tau_1} k$$

which is the expected dispersion relation for wave propagation with a speed corresponding to

$$c_0 = \frac{1}{\sqrt{2}} \lim_{\epsilon \rightarrow 0} \frac{\lambda}{\tau}$$

Correction to this dispersion relation vanish at the order $O(\epsilon^2)$ and we obtain a correct wave dispersion relation corresponding to what was found with equations (2.34)-(2.36):

$$\begin{aligned} O(\epsilon^2) : \quad i\omega\tau_2 &\Leftrightarrow \frac{\omega^2}{2} \tau_1^2 = \Leftrightarrow \frac{\lambda_1^2 k^2}{4} \\ \text{thus} \quad i\omega\tau_2 &= \frac{\lambda_1^2 k^2}{4} \Leftrightarrow \frac{\lambda_1^2 k^2}{4} = 0 \end{aligned}$$

Of course, in finite system, corrections of higher order may show up as may be seen on figure 2.5.

Up to here, we have retrained our direct derivation of a dispersion relation to the case $4\mathcal{N}2$. On $5\mathcal{N}2$, $W = W(n)$ is the full symmetrized matrix, with n the

indices of refraction. The matrix $W(n)$ is obtained from (2.47) after a change of variable given by (2.51) corresponding to the introduction of a special mass m_0 for the rest particles: $m_0 = \sqrt{Nn^2 \Leftrightarrow N}$; now if $N = 4$ we obtain finally

$$W = \frac{1}{2n^2} \begin{pmatrix} 2n^2 \Leftrightarrow 4 & 2\sqrt{n^2 \Leftrightarrow 1} & 2\sqrt{n^2 \Leftrightarrow 1} & 2\sqrt{n^2 \Leftrightarrow 1} & 2\sqrt{n^2 \Leftrightarrow 1} \\ 2\sqrt{n^2 \Leftrightarrow 1} & 1 & 1 & 1 \Leftrightarrow 2n^2 & 1 \\ 2\sqrt{n^2 \Leftrightarrow 1} & 1 & 1 & 1 & 1 \Leftrightarrow 2n^2 \\ 2\sqrt{n^2 \Leftrightarrow 1} & 1 \Leftrightarrow 2n^2 & 1 & 1 & 1 \\ 2\sqrt{n^2 \Leftrightarrow 1} & 1 & 1 \Leftrightarrow 2n^2 & 1 & 1 \end{pmatrix} \quad (2.56)$$

The dispersion relation is given by $\exp(i\omega\tau) = 1$ or root of

$$n^2 X^4 + (2 \Leftrightarrow \kappa) X^3 + 2(1 \Leftrightarrow \kappa \Leftrightarrow (1 \Leftrightarrow n^2) \cos(\lambda k_1 + \lambda k_2)) X^2 + (2 \Leftrightarrow \kappa) X + n^2$$

One observe the same relation as before (2.53) if $n = 1$. The general solution may be found after some algebra, but going here into this derivation is out of purpose, so we entrust the task to the reader.

2.2.3 TLM versus the finite difference scheme

The reader may now ask a legitimate question: why is the LB approach for wave propagation better than a finite difference scheme directly applied on the wave equation? Indeed the direct discretisation of the wave equation is straightforward and we may worry about the progress made so far. This subsection is devoted to a comparison between the LB wave method and the direct discretisation of the wave equation. The crucial advantage of the LB method will best show up only while evaluating its ability to simulate various real physical situations. Indeed, the power of the approach is particularly evident in the management of complex boundary conditions inherent to most real phenomena. However, qualitative differences are yet visible from a more fundamental point of view.

The discrete wave equation is

$$\frac{f(\vec{r}, t + \tau) + f(\vec{r}, t \Leftrightarrow \tau) \Leftrightarrow 2f(\vec{r}, t)}{\tau^2} = c^2 \frac{\sum_i f(\vec{r} + \vec{v}_i \tau, t) \Leftrightarrow 4f(\vec{r}, t)}{\lambda^2}$$

Finding solutions in the form of a plane wave on a regular square lattice

$$f = \exp(i(\vec{k}\vec{r} \Leftrightarrow \omega t))$$

gives the dispersion relation

$$\cos(\omega\tau) \Leftrightarrow 1 = c^2 \frac{\tau^2}{\lambda^2} (\cos(k_1 \lambda) + \cos(k_2 \lambda) \Leftrightarrow 2)$$

If we apply the same multi-scale method as before we obtain for the order $O(\epsilon^0)$:

$$O(\epsilon^0) : \quad \omega^2 = c^2 k^2$$

which is the expected dispersion relation. But now, collecting the terms of the next order we have:

$$O(\epsilon^2) : \quad \omega^2 = c \frac{\lambda_1}{\tau_1} \sqrt{k_1^4 + k_2^4} \quad (2.57)$$

Which is wrong and anisotropic. In conclusion, anisotropic deviations from the expected relation show up at the order ϵ^2 already. A good way to appreciate the anisotropy is to write down the dispersion relation for a specific orientation ϕ of the wave vector \vec{k} with respect to the lattice orientation (corresponding to the coordinates system orientation). So let \vec{k} be defined by $k_1 = |k| \cos(\phi)$ and $k_2 = |k| \sin(\phi)$. The second order term of the dispersion relation (2.57) becomes

$$O(\epsilon^2) : \quad \omega^2 = c \frac{\lambda_1}{\tau_1} k^2 \sqrt{\cos^4(\phi) + \sin^4(\phi)}$$

If $c = \lambda_1/\tau_1$ then we have a correct dispersion relation (up to the second order in ϵ) for $\phi = 0$, whereas corrections show up for any other orientations ϕ . Another example is if $c = \lambda_1/\sqrt{2}\tau_1$, this time we rescue the dispersion relation on the particular orientation $\phi = \pi/4$ because

$$\phi = \frac{\pi}{4} \quad \Rightarrow \quad \sqrt{\cos^4(\phi) + \sin^4(\phi)} = \frac{1}{\sqrt{2}}$$

In the LB method those kind of anisotropic errors, where a specific ϕ allows a supplementary correct order of magnitude for ϵ , appears at the level $O(\epsilon^3)$. This conclusion provides one first inside in the difference between our LB method and a more straightforward discretisation of the wave equation. But as mentioned at the beginning of this sub-section, the very advantage of the method will definitively show up as boundary conditions will be introduced for simulating real problems.

2.3 Generalization of the linear approach

So far, we have constructed our wave automaton on the paradigm presented in equations (2.1), (2.2), (2.3). The two last equations express the conservation of two quantities: a scalar I and its current \vec{J} . The first, constitutive equation fixes the response of the medium to the presence of fields; its form was deduced from similarities between different physical equations exhibiting wave phenomena. Is it possible to think of a more general constitutive equation? The form of the constitutive equation could be well deduced or constructed from the linearity and the symmetries alone - that we chose to plug in our model. Imposing only linearity into account and writing only the constant and first order derivative terms, the most general stress tensor $\Pi_{\alpha\beta}$ reads

$$\Pi_{\alpha\beta} = \chi_0 I \delta_{\alpha\beta} + \chi_1 \partial_t I \delta_{\alpha\beta} + \chi_2 \partial_\gamma J_\gamma \delta_{\alpha\beta} + \chi_3 \partial_\alpha J_\beta + \dots$$

where the χ_l are coefficients characterizing the medium. Now, the requirement of time reversal invariance means that under the transformation $t \rightarrow t' = \Leftrightarrow t$, the related physical quantities transform in a consistent fashion so that the *form* of the equations is the same as before. As a consequence of the continuity equation (2.2) defining \vec{J} , \vec{J} must transform as follows: $\vec{J} \rightarrow \vec{J}' = \Leftrightarrow \vec{J}$. This is also intuitively obvious. Consequently, to maintain the form of the constitutive equation one must cancel all the terms but the first in the expression of Π given above. This is precisely the form we have chosen: $\Pi = \chi_0 I \delta_{\alpha\beta}$. Note that, as expected, the form of the cancelled terms is exactly that of the dissipative terms eliminated by the choice $\xi = 1/2$ in (2.33). In addition, the time reversal invariance is here equivalent to the requirement of locality, in space and time. Locality because the response of the medium, namely the constitutive equation, contains no space- or time-derivative. This means that the medium responds instantly and linearly to the traveling perturbation. Thus, the set of equations (2.1)-(2.3) given in the introduction of this chapter, has a taste of universality.

Looking for the most general propagation matrix

In this section we want to exhibit the algebraic consequence of the conditions imposed by our paradigm in order to systematically construct the most general possible propagation matrix in \mathbb{C} . This will allow a deep comparison with the approach of other authors [27, 28, 29] starting from a similar but different point of view.

The discretisation of our model is based on the LBGK paradigm since the dynamics is expressed as a relaxation towards a state of local equilibrium. In general, a (linear) state of local equilibrium may be defined with a projector P applied on the set of f_i 's:

$$P\mathbf{f} = \mathbf{f}^{(0)}$$

where the flux are now complex valued quantities. The detailed form of P is let aside by now, but, by definition of the equilibrium, we must clearly have $P^2 = P$. Now W reads

$$W = \frac{1}{\xi}P + \frac{\xi \Leftrightarrow 1}{\xi}\mathbf{1}$$

where $\mathbf{1}$ is the identity matrix. P is idempotent, so the eigenvalues of P are all either 0 or 1 and consequently the eigenvalues λ_W of W are

$$\lambda_W \in \left\{ \frac{\xi \Leftrightarrow 1}{\xi}, 1 \right\}$$

At this point we have as many independent conserved quantities as the multiplicity of the eigenvalue $\lambda_W = 1$. So, in our case, P must be chosen in order to have 3 eigenvalues of modulus 1. A basis of the corresponding 3-dimensional

eigenspace E_0 : \mathbf{e}_I , \mathbf{e}_{J_x} and \mathbf{e}_{J_y} must exist and we can use them to define the conserved scalar quantities.

$$\begin{aligned} I &\doteq \bar{\mathbf{e}}_I^\perp \mathbf{f} \\ J_x &\doteq \bar{\mathbf{e}}_{J_x}^\perp \mathbf{f} \\ J_y &\doteq \bar{\mathbf{e}}_{J_y}^\perp \mathbf{f} \end{aligned}$$

We write E_\perp the eigenspace orthogonal to E_0 , namely corresponding to $\lambda_W \neq 1$. States in E_0 are closed under inversion of space, namely R has the property that

$$\mathbf{f} \in E_0 \quad \Rightarrow \quad R\mathbf{f} \in E_0$$

Now, the time reversal invariance condition reads

$$WR\bar{W}\mathbf{f} = R\mathbf{f} \quad (2.58)$$

where \bar{W} is the complex conjugate of W . If $\mathbf{f} \in E_\perp$ one can deduce from (2.58) the following condition on ξ :

$$\begin{aligned} W\mathbf{f} &= \frac{\xi \Leftrightarrow 1}{\xi} \mathbf{f} \quad \Rightarrow \\ \frac{\bar{\xi} \Leftrightarrow 1}{\bar{\xi}} \frac{\xi \Leftrightarrow 1}{\xi} &= 1 \quad \Rightarrow \end{aligned} \quad (2.59)$$

$$\xi \doteq \frac{1}{2} e^{i\phi} \quad (2.60)$$

where ϕ represents a possible phase for ξ . Because $RWR = W$ from the space inversion invariance condition (see (2.48)) and $WR\bar{W} = R$ from the time reversal invariance condition then $W\bar{W} = 1$, namely $|\lambda_W| = 1$, so that (2.59) reads

$$\left(1 \Leftrightarrow 2 \exp(i\phi)\right) \left(1 \Leftrightarrow 2 \exp(\Leftrightarrow i\phi)\right) = 1 \quad \Rightarrow \quad \phi = 0$$

Thus $\xi = 1/2$ is real-valued; note that we found the same value as before. Finally the eigenvalues of W , are real: $\lambda_W = \pm 1$. It is easy to check that this is a sufficient condition to respect the expected symmetries. At this stage we have not necessarily the unitarity of W . For our model unitarity is not required by the paradigm, however non-unitary propagation matrices must be avoided since they allow indefinitely large growth of $|\mathbf{f}|$ and this may show up numerical instabilities. Suppose our matrix W is symmetrizable then we can make a change of variable by mean of M and work with the symmetric matrix $W' = M^{-1}WM$ as done in (2.50). Then

$$W\bar{W} = 1 \quad \Rightarrow \quad W'\bar{W}' = 1 \quad \Rightarrow \quad \bar{W}'^\perp = 1$$

So W' is unitary and this avoid the problem of indefinitely large growth of $|\mathbf{f}|$. Consequently, acceptable propagation matrix W must be symmetrizable. We are

not allowed to chose any matrix M because besides the propagation matrix W is hidden the effective streaming phase which actually moves the f_i 's according to their direction v_i 's. Consequently the possible change of variable that preserve the form of the streaming operator are diagonal: $M_{ij} \doteq m_i \delta_{ij}$. Thus, the question is: what are the conditions on our model under which W is symmetrizable by a diagonal change of variables? W is symmetrizable if P is, thus, with the detailed form definition of P given by (2.9), we have

$$\begin{aligned} (M^{-1}PM)_{ij} &= (M^{-1}PM)_{ji} \\ \frac{a_i}{m_j} m_i + \frac{b_i v_{i\alpha} v_{j\alpha} m_j}{m_i} &= \frac{a_j}{m_i} m_j + \frac{b_j v_{i\alpha} v_{j\alpha} m_i}{m_j} \\ m_j^2 (a_i + b_i v_{i\alpha} v_{j\alpha}) &= m_i^2 (a_j + b_j v_{i\alpha} v_{j\alpha}) \\ \Rightarrow m_j &= m_0 \sqrt{\frac{a_i}{a_0}} \end{aligned}$$

This solution allows us to symmetrize P , providing that $b_i = b$. Moreover for isotropy reasons we must have $a_{i \neq 0} = a$ and it is thus possible to set m_0 so that $m_{i \neq 0} = 1$. This is actually what was done in (2.51) but the context is here more general. The presence of different b_i avoids the symmetrization of P and lattices requiring different b_i , namely lattices with different velocities are consequently spurious; this may well explain the instabilities observed for the case $8\mathcal{N}2$. So unitarity and symmetry are two necessary but not sufficient conditions for our model. It is interesting to note that these conditions are exactly the starting point of a wave automaton derivation presented by Christian Vanneste *et alter* in [27, 28, 29]. Their conditions (unitarity and symmetry) were directly deduced from the requirement of energy, namely $|\mathbf{f}|^2$, conservation and time reversal invariance. W is considered as a scattering matrix, named S hereafter, and isotropic consideration implies that the two transmissions, the four reflections and the four rotations are equivalents to each-other so that the most general symmetric in $4\mathcal{N}2$ reads

$$S = \begin{pmatrix} te^{i\alpha} & de^{i\beta} & re^{i\gamma} & de^{i\beta} \\ de^{i\beta} & te^{i\alpha} & de^{i\beta} & re^{i\gamma} \\ re^{i\gamma} & de^{i\beta} & te^{i\alpha} & de^{i\beta} \\ de^{i\beta} & re^{i\gamma} & de^{i\beta} & te^{i\alpha} \end{pmatrix} \quad (2.61)$$

$$\overline{S}^\perp S = 1 \quad \Rightarrow \quad \begin{cases} t^2 + r^2 + 2d^2 & = 1 \\ rt \cos(\alpha \Leftrightarrow \gamma) + d^2 & = 0 \\ r \cos(\gamma \Leftrightarrow \beta) + t \cos(\alpha \Leftrightarrow \beta) & = 0 \end{cases} \quad (2.62)$$

where r, t, d, α, β and γ are real valued quantities. Together with the unitarity conditions it remains 3 parameters. In other words the magnitude of the different scattering configurations may be chosen; but the phase is then imposed in order

to preserve energy conservation. Our approach differs in the conditions added to unitarity and symmetry because the eigen vectors are given by the definitions of the physical quantity. The difference between the two approaches is that our LB model conserve a quadratic form (essentially for numerical reasons) and 3 specific linear forms (corresponding to the physical relevant quantities: I and its current \vec{J}) in the phase space of the automaton. The most general symmetric propagation matrix in $4\mathcal{N}2$ that respects the form of the eigenvectors $M\mathbf{e}_{[I,Jx,Jy]}$, reads

$$W = \frac{2}{4 + m_0^2} \begin{pmatrix} \frac{m_0^2}{2} \Leftrightarrow 2 & m_0 & m_0 & m_0 & m_0 \\ m_0 & 1 & 1 & \Leftrightarrow \frac{m_0^2}{2} \Leftrightarrow 1 & 1 \\ m_0 & 1 & 1 & 1 & \Leftrightarrow \frac{m_0^2}{2} \Leftrightarrow 1 \\ m_0 & \Leftrightarrow \frac{m_0^2}{2} \Leftrightarrow 1 & 1 & 1 & 1 \\ m_0 & 1 & \Leftrightarrow \frac{m_0^2}{2} \Leftrightarrow 1 & 1 & 1 \end{pmatrix} \quad (2.63)$$

$$\overline{W}^\perp W = 1 \quad \Rightarrow \quad m_0 \in \mathbb{R}$$

It is straightforward that the unitarity condition imposes real-valued m_0 so that we found the same matrix as before in (2.56). In conclusion our generalized propagation matrix is real with one parameter allowing locally different propagation speed. Note that (2.61) corresponds to (2.63) with $m_0 = 0$ providing $\alpha = \beta = \gamma$. Our approach takes naturally into account the fundamental properties defining the propagation of waves: conservation of energy, energy flux in a linear, isotropic and elastic medium. Moreover we have shown how to break the time reversal invariance and simulate “viscous” medium on hexagonal 2-dimensional lattices.

The presence of 3 parameters in the scattering approach (2.61) indicates that the model developed by Vanneste possibly constitutes a more general approach than our LB model. However this generalization, issued from a transmission line paradigm seems not to correspond to a physical continuum situation, at least in the classical approach. Indeed, if we consider, for instance, the Maxwell equations for macroscopic (linear, isotropic and elastic) media[18], in the absence of free charges or current, we can derive conservation laws for both energy *and* linear momentum (or energy flux) leading to the set of equations already mentioned in the preamble of this chapter. In Vanneste’s derivation [28], the conserved quantity $\vec{f}^\perp f$ is called energy flux but we believe that this expression represents the energy *magnitude* rather than the energy flux, since a flux or momentum is a first order tensor (vector) on the velocity space (like our \vec{J} defined in (2.8)). Consequently, the conservation of a vectorial energy flux is not treated in Vanneste’s derivation and this may explain the apparent possible additional choice of their model.

Consequently, the scattering approach is not a generalization of our LB wave model. Indeed, a first attempt to conserve a vectorial flux of energy in the

scattering model would consist to impose the conservation of

$$\text{vectorial energy flux} = \begin{pmatrix} f_1 \bar{f}_1 \Leftrightarrow f_3 \bar{f}_3 \\ f_2 \bar{f}_2 \Leftrightarrow f_4 \bar{f}_4 \end{pmatrix}$$

but this is incompatible with the unitarity imposed on the matrix (2.61). However, it turns out that the scattering approach can be interpreted as a generalization for complex states f 's of the LB model for the diffusive propagation. This model [30] has been studied intensively and some of its applications were mentioned in the introduction. The matrix formulation of the Boltzmann model for the diffusion equation given in (A.7) takes a form very similar to (2.61) (corresponding to the case $\alpha = \beta = \gamma = 0$) with a conservation condition which simply reads $r + t + 2d = 1$ instead of (2.62). Consequently, we believe that the Vanneste's approach is equivalent to define a complex random walkers whose probability of presence on the i 'th link is $|f|^2$.

Algorithms & Implementations

3.1 Introduction

The power of a numerical model, besides its ability to qualitatively or quantitatively mimic a particular phenomenon, relies on its simplicity and performance once it has been implemented on a computer, namely once it has come true out of a scientific mind. Simplicity by itself is a rather “philosophical” requirement that is shared, or should be shared by the entire scientific community because it is considered as the essential property of something well understood; A. Einstein will not mind with his motto: “Everything should be made as simple as possible, but not simpler”. In our case, the simplicity of the Lattice Boltzmann wave model is the key point that allows fast computation and thus permits the implementation of very efficient numerical tools for intensive theoretical experiences and/or numerical simulations.

Our LB approach for wave propagation is computationnaly simple because it is working with regular data structures, with only local calculation and nearest neighbors communications. Consequently, it is well adapted to Massively Parallel Processors (MPP), or every Single-Instruction-Multiple-Data (SIMD) purpose machines in general. As we will show later on in the next chapter, this does not mean that other architectures such as Multiple-Instruction-Multiple-Data (MIMD) purpose machines, sequential or vector computers cannot host efficiently our algorithm.

This section describes in some detail the kernel of our algorithm that is generic to various applications developed later in this chapter. This kernel has been used as a benchmark for regular scientific parallel applications and we shall show the performances obtained on a wide range of different computers. Finally, the paral-

lization (the global parallelism is naturally embeded in the synchronous motion of the flux, see chapter 2) of the code in various programming models will be studied, involving some considerations on the compromise between performance and coding effort. For the reader who may be interested, please note that a special effort was done by P. Kuonen *et al.*[31] at the Swiss Federal Institute of Technology (EPFL) to provide an object-oriented approach of our algorithm.

3.2 LB Wave Automata Kernel: A New Benchmark

The sequential kernel of any application using our LB method to model wave propagation is based upon the version given in figure 3.1. This kernel is sufficiently simple to be rewritten in a suitable language to fit the best compiler available on a specific architecture. Our algorithm is customized to model radio wave propagation in urban cells as described in section 4. The generic problem is that of a source (an antenna) placed somewhere in a discretized portion of an urban area. One simulates the propagation of the waves emitted by this antenna until the waves reach the boundary of the domain. Finally one computes the intensity patterns in the layout to predicted, for instance, the coverage of the antenna. Some variables will only be justified later in section 4. Basically, the algorithm corresponds - firstly - to apply the 4×4 propagation matrix from equation 2.52 of the previous chapter at every site, with the introduction of a source and special coefficients to account for various boundary conditions, and - secondly - to implement the spreading phase which moves the data across the system.

Figure 3.1 corresponds to the operations involved during one iteration. For simplicity, the syntax used is that from Fortran90. Fortran90 is the most widely available and accepted data parallel language. It extends Fortran77 with a variety of new constructs such as pointers, including arrays and operations that are particularly suited to data parallel problems in which the data can be represented by arrays. Fortran90 provides a number of basic data parallel functions as built-in array operators and intrinsic functions. It also provides constructs, such as the `where` and the `forall`, which assist in programming more complex data parallel functions. The Fortran 90 array extensions are suitable for efficient implementation on both SIMD and MIMD parallel machines and commercial implementations are available. The earliest language subsets available on parallel machines was CM Fortran by Thinking Machines, which has compilers for the CM2 and the CM5.

We shall next go through the program lines given in 3.1. The variables declaration is given here without comment, except that, even if it is not explicitly mentioned in the figure, the variables are considered as dynamically allocated to

```

SUBROUTINE lbwave(i)

  INTEGER i                               ! Iteration time
  INTEGER nx,ny                           ! Size of the arrays
  INTEGER sx,sy                           ! Position of the source
  REAL    g                               ! Amplitude of the source
  INTEGER t                               ! Period of the source

  REAL,   DIMENSION(1:nx, 1:ny):: if1,if2,if3,if4 ! f_in: before collision
  REAL,   DIMENSION(1:nx, 1:ny):: of1,of2,of3,of4 ! f_out: after collision
  REAL,   DIMENSION(1:nx, 1:ny):: c1,c2         ! Two medium coefficients
  REAL,   DIMENSION(1:nx, 1:ny):: p           ! Value of the field
  REAL,   DIMENSION(1:nx, 1:ny):: a2         ! Integration variable
  INTEGER,DIMENSION(1:nx, 1:ny):: d         ! Delay variable

1 : WHERE((if1.ne.0).or.(if2.ne.0).or.(if3.ne.0).or.(if4.ne.0))

2 :     d   = d + 1
3 :     p   = if1+if2+if3+if4
4 :     a2  = a2 + p**2

5 :     of1 = c1*( c2*p - if3 )                ! The collision term
6 :     of2 = c1*( c2*p - if4 )
7 :     of3 = c1*( c2*p - if1 )
8 :     of4 = c1*( c2*p - if2 )

9 : END WHERE

10: of1(sx,sy) = g*SIN(i*2*PI/t)              ! The source
11: of2(sx,sy) = g*SIN(i*2*PI/t)
12: of3(sx,sy) = g*SIN(i*2*PI/t)
13: of4(sx,sy) = g*SIN(i*2*PI/t)

14: if1       = CSHIFT(of1,DIM=1,SHIFT=-1)    ! The streaming
15: if3       = CSHIFT(of3,DIM=1,SHIFT= 1)
16: if2       = CSHIFT(of2,DIM=2,SHIFT= 1)
17: if4       = CSHIFT(of4,DIM=2,SHIFT=-1)

  END SUBROUTINE

```

FIGURE 3.1: This is the non-optimized algorithm, written in F90, for one iteration of the 4×4 LB wave model. Actually it represents the application kernel to radio wave propagation simulation presented in section 4 and various definition are referring to this section. Our benchmark is directly derived from this algorithm but is written in C to allow comparison on a wide range of machines.

allow a full adaptability of the algorithm. More precisely, we have:

line 1 This the so-called “causality” context which avoids computation of no use. Indeed the simulation starts with 0-valued f 's and informations spread step by step out of a source location.

line 2 The parallel variable d , counting the iterations a site is visited, is incremented. This enters the calculation of the distance to the source which is important for renormalizing our results in the context of radio wave propagation prediction (see 4.2). $d = 0$ on the source at time $i = 0$.

line 3 The value of p , the relevant “macroscopic variable” corresponding to the I of the theory given in chapter 2 is calculated and ...

line 4 ... its square is integrated for later extraction of the amplitude of the local oscillation.

line 5-8 The new f 's are calculated as a linear combination of the old f 's. For the case $c_1 = 1/2$ and $c_2 = 1$, this corresponds to the TLM matrix given in 2.52. Other values of c_1, c_2 , which are arrays of the same size than the dimension of the system, may be locally set in order to account microscopically for various boundary conditions.

lines 11-14 A sinusoidal source defined by its period and amplitude is imposed for all the f 's on a specific site.

lines 15-17 The data are moved across the lattice corresponding to the direction of propagation usually associated to the f 's.

It is interesting to note that lines 5-8 are not the exact transcription of the matrix 2.52: on the one hand, the use of the quantity $p=f1+f2+f3+f4$ allows a reduction of the operations performed per iteration, and on the other hand, the computations are made purely local; the data motion is grouped in a separate set of instructions: line 14-17. Clearly, a further reduction of the number of operations is possible if we compute $p=c2*p$ once before the lines 5-8. Finally only 15^1 floating-point operations (Flop) for each site in the causality context are necessary if we neglect the extra computation needed on the source location. The number of sites updates and thus the total number of Flop depend on the number of iterations and the size of the system. Suppose we place the source in the middle of the array and we chose to perform n iterations on a system size $n \times n$. As we shall see later on (chapter 4), this is enough (but not too much) for the purpose of predicting the coverage of the antenna in the surrounding urban cell. The causality context is growing between each time steps so that some

¹More precisely, it is 14 floating-points operations plus 1 integer addition but we adopt 15 for simplicity.

algebra (to be found later, in section 4.5) is necessary to derive the total number of site-updates performed during n iterations on a system size $n \times n$. Finally we have

$$N_{\text{Flop}} = 15 \frac{d^3 + d}{2}$$

The memory used by our algorithm is critical since it grows like the square dimension of the domain which must be as big as possible. It is possible to reduce the number of full arrays, namely $n \times n$ arrays, from 13 to 8 by introducing a temporary variable `temp` and by suppressing the distinction in/out. Meanwhile no extra Flop is added but we must abandon the array features and notations typical of F90 and return to the F77 notation with indices. Focusing on the computation efficiency, lines 2-8, are thus replaced by:

```

d(i,j)  = d(i,j)+1
p       = f1(i,j)+f2(i,j)+f3(i,j)+f4(i,j)
a2(i,j) = a2(i,j) + p * p

p       = c1(i,j)*p

temp    = c2(i,j)*( p - f3(i,j) )
f3(i,j) = c2(i,j)*( p - f1(i,j) )
f1(i,j) = temp

temp    = c2(i,j)*( p - f4(i,j) )
f4(i,j) = c2(i,j)*( p - f2(i,j) )
f2(i,j) = temp

```

Let us consider now the section where the data are moved : line 14-17. This section requires as much attention as the previous calculation section. Indeed, for the optimized benchmark version of the code presented below, about half of the time was spent in communication. The “communication” time denotes the time spent to move the data across the array but not the time needed for accessing the memory. We measure this ratio by comparing execution times with and without lines 14-17. The first possible improvement is to rewrite the code without the array instruction `cshift` using a full indices notation. But the best solution was found in the C programming language with the function `memmove`² which copies as efficiently as possible a number of bytes from one memory area to another; it does not check for overflow of any receiving memory area, but copying between objects that overlap will take place correctly.

`memmove` associated with our pointers structures allows us to move, and move efficiently, only half of the data compared to a straightforward implementation of

²From the man-pages: `void *memmove(void *s1, const void *s2, size n)` copies `n` bytes from memory areas `s2` to `s1`.

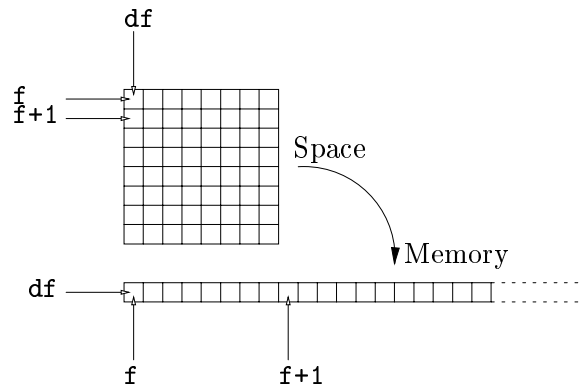


FIGURE 3.2: *Memory representation of the data arrays. The arrays are “built” in memory via pointer to the first element of each line: a line always corresponds to a contiguous block of memory.*

a `cshift` equivalent operation. Indeed, each array of data is allocated as a linear block of memory and let us write `df` a pointer to the first element of this array; see figure 3.2. We then built an array of pointers to the lines of the data array: `f` so that `f[i][j]` represents, in the C syntax, the value of the array at line `i` and column `j`. Now, remember that, following the usual directions labels given in figure 2.1, `df1` and `df3` will always move across the columns whereas `df2` and `df4` will move across the lines. Consequently, the actual motion of the data in `df2` and `df4` is achieved by moving only the pointers `f2` and `f4` respectively so that lines 14-17 may take the following form:

```

memmove(f1[0] , f1[0]+1, (nx*ny-1)*sizeof(float));
memmove(f3[0]+1, f3[0] , (nx*ny-1)*sizeof(float));

pt=f2[dim-1];
memmove(f2+1, f2, (ny-1)*sizeof(float *));
f2[0]=pt;

pt=f4[0];
memmove(f4, f4+1, (ny-1)*sizeof(float *));
f4[dim-1]=pt;

```

In the `memmove` approach, we have displaced only about half of the bytes in the memory. It is also possible to act on `f1` and `f3` the same way as performed on `f2` and `f4` and thus make only pointer motion and no real data movement. But we must first transpose `f1` and `f3` and consequently the inner loop no longer access successively contiguous values in the memory. It turns out that this, together with the extra indices calculation involved by the transposition is not that much profitable.

Processor	Sun Ultra 1	SG RISC 10000	IBM RISC 6000	Intel Pentium Pro
Frequence	167 MHz	175 MHz	66 MHz	200 MHz
Memory	124 MB	1125 MB	192 MB	64 MB
Time	142 s	90 s	100 s	120 s
Performance [MFlop/s]	11.4	18	16.2	13.5

FIGURE 3.3: *Sequential performance of the LB wave benchmark on different machines. The code, shown in appendix B, was written in C and compiled with usual optimization option `-O3` or `-fast`. It was run during 600 iterations for a system of size 600×600 ; the executable size is about 12 Mega bytes. We have counted 1 Flop for each multiplication or addition, so 15 Flop per site update. MFlop/s means millions of Flop per second.*

It is important to note that a `memmove` may directly replace a `CSHIFT` only if the values on boundaries of the arrays - after the calculation - are homogeneous. Indeed, during a `memmove` operation, the “left” boundary site of each line i will be replaced by the “right” boundary site of line $i \pm 1$. For us it is not a problem - as we shall show in section 4.1 - since $\mathbf{f}=0$ at any time on the boundary (boundary absorption).

Consequently, following the remarks given above, we are now able to write down an optimized algorithm for the kernel of our LB wave model. It is written in C and the details of the code are given in appendix B. It was run on different computers commonly available at the publication date of this dissertation and the performances are summarized on table 3.3; we have used the most common optimization option, *i.e.* `-fast` or `-O3`.

The scientific community is more used to Fortran language and one may ask why C was chosen. Fortran, and especially Fortran 90 and its array features is evidently more comfortable for developers of scientific applications, but we have observed that: (i) identical performances were obtained with or without dynamical memory allocation (ii) array features such as `cshift` are very inefficient and rewriting the data motion with indices allows a global gain of performance of about 25% and (iii) only the full Fortran 77 was a little more efficient than the C code; all other F90 attempts gave dramatically poor performances: see table 3.4.

In conclusion we can be satisfied with the effort in optimizing and rewriting the algorithm of our kernel since it we have improved our performance by a factor of 2.27. It was worth the effort especially because of the simplicity and the small size of the code. But typical simulation still takes more than 2 minutes of time, even on powerful computers, so more performance is needed to built real interactive tool (with all the extra time involved by the visualization interface). This excess of computational power will be obtained in by a careful parallelization

F77	C	F90 (no <code>cshift</code>)	F90
1.07	1	0.57	0.44

FIGURE 3.4: Comparison between C, F77 and F90 for our LB wave kernel. “no `cshift`” indicates that full indices notations were used to implement the data motion. We show the ratio of the Flop/s relative to our C benchmark. The measurements were performed on an UltraSparc 1 (125 Mbytes).

of our model, as will be shown in the next section.

3.3 Parallelization

In conventional computing, a computer has a single central processor, which operates sequentially on data. If a list of operations is to be performed on many data elements as is the case with our model, the computer must still perform the operations separately on each element, one after another.

Many problems, especially many scientific ones, have large amounts of data, and solutions to the problem can be described in terms of what to do to each element of the data, or more generally, to subsets of the data. These kinds of problems have been termed *data parallel* because the operations can often be applied to elements or subsets of the data at the same time, and thus efficient computations are possible on a parallel computer.

The parallelization of our code is a necessary step towards fast simulations on very large systems. Parallelism is one of the presently best computational strategy to run faster complex or large tasks. It is done by splitting the algorithm and/or the data into smaller parts and assigning them to multiple processors to work on simultaneously. As a collective behavior leading to a global solution, parallelism relies on the processors ability to communicate to achieve coordination. The communication between processors depends upon memory architecture, which, in turn, will affect the way a parallel program is written. The three primary memory architectures are shared memory, distributed memory and memory hierarchies.

In a shared memory architecture multiple processors operate independently but share the same memory resources, namely one global address space is available. Only one processor can access the shared memory location at a time and synchronization is achieved by controlling tasks’ reading from and writing to the shared memory. In the distributed memory architecture, multiple processors operate independently but each has its own private memory. Data is shared across a communications network using message passing and the user is responsible for synchronization also using message passing. The hierarchic memory architecture combines several level of shared memories within a group of processors and between groups.

There are many methods of programming parallel computers. Among them we have investigate, and applied to our benchmark, the data parallel-, the message passing- and the shared memory-programming models. Note that these models are machine/architecture independent, any of the models can be implemented on any hardware given appropriate operating system support. It turns out that an effective implementation closely matches its target hardware and provides the user ease in programming and good performance.

In order to provide a performance analysis of our algorithm which is independent of the problem size $n \times n$ and the number of processors p involved, we will analyze the execution time of n iterations T_p according to the following simple scaling law:

$$\frac{T_p}{n} = a \frac{n^2}{p} + bn \quad (3.1)$$

In this expression, a represents the time unit of a site calculation and thus $1/a$ is the effective number of Flop per second for one processor. The term bn accounts for the inter-processor communication and thus $1/b$ is proportional to the bandwidth of the whole machines. This means implicitly that the time spent during communication only depends on the size of the message: this is justified in our case because relatively few large messages are sent across the network. Indeed the general principle of our parallelization is based upon a domain decomposition across the processors and it involves at each time steps communication of the boundaries. Consequently T_p/n^2 is a linear function of n/p with a slope a and constant b . This model will be used throughout the remaining part of this section where we will explore different programming models applied to our benchmark.

3.3.1 Data Parallel Programming Model

Data parallelism is possible when a sequential list of instruction are acting upon a data structure in parallel and transparently. In data parallel computing, there are many processors, and each data element is associated with a processor. All processors can therefore perform the same operation on their data at the same time. If there are more data elements than there are processors (which is usually the case), the system creates virtual processors by dividing up the memory associated with each physical processor. All communications are hidden to the programmer who sees the effectively distributed memory as a single block.

Programming with data parallel model is very simple since it avoids the complexity of trying to solve a naturally parallel problem in a serial manner. It is accomplished by writing a program with data parallel constructs such as array-handling facilities from Fortran 90 and compiling it with a data parallel compiler such as HPF, for instance. The kernel is very similar to the starting point of the optimization process of the preceding section given in the figure 3.1 but compiler directives must be added to tell the operating system how to distribute the data

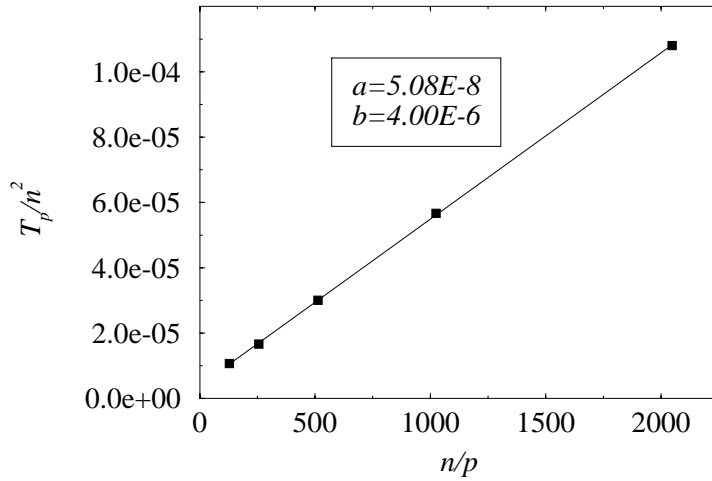


FIGURE 3.5: Performance analysis of the LB Benchmark for the CM200 (256 FPU). The execution time T_p fits perfectly to our performance model given in equation 3.1. The CM was considered as a whole, namely $p = 1$.

across the processors. This distributing phase is one of the key step of the process and will strongly determine the performance of the implementation.

Note that, in the data parallel model, the so called “causality” context (line 1) is actually not dividing by 2 the total number of operations. Indeed it is idle effort to avoid useless calculation on empty nodes, since the processors in charges of these nodes must anyhow wait for the others to complete their operations before moving to the next step. Indeed, there is an implicit synchronization due the communication. This is a consequence of the present impossibility - present compilers don’t do it - in data parallel programming, to dynamically redistribute the data across the processors to achieve a better load balancing.

The result obtained with the Connection Machine CM200 is presented in figure 3.5. The CM system is an integrated combination of hardware and software designed for high-speed data parallel computing. The programming language is the CMFortran (CMF), an extended implementation of the Fortran77 and precursor of fortran90. The execution time T_p fits perfectly our performance model given in equation 3.1. The overall performance of the CM200 is high since we obtain for a complete simulation (n iterations on $n \times n$ arrays) a site update frequency of nearly 18 MHz.

However some dramatic limitations must be taken into account while appreciating these high performances. The arrays must be statically allocated with dimensions in power of 2, and the size of the arrays are strictly limited by the physical memory (no swapping) of the parallel processing unit with a maximum size of 2048×2048 on a CM200 with 256 Mbytes.

3.3.2 HPF, The New Standard

Today data parallel computing may be done on more general computers by means of the High Performance Fortran (HPF). HPF is the only data parallel standard, it is based on Fortran 90 and looks very similar to CMF. The most important compilers directives, namely those we have used in our benchmark, are as follows:

```
!HPF$ PROCESSORS p(NUMBER_OF_PROCESSORS())
```

which declares processors arrangements `p` onto which objects can be distributed. The intrinsic function `NUMBER_OF_PROCESSORS` returns the number of available processors (*i.e.* given by the operating system of run-time specified by the user) for a particular execution. In the example above, all the available processors are here logically arranged as a 1-dimensional array. Other arrangements like 2-dimensional array of processors are also possible.

```
!HPF$ DISTRIBUTE f(*,BLOCK) ONTO p
```

which specifies a data mapping of a 2-dimensional array `f` to the abstract processor arrangement `p`. The expression `(* ,BLOCK)` means the second dimension is distributed by block on the available processors.

```
!HPF$ ALIGN g(:, :) WITH f(:, :)
```

specifies that a data object is to be mapped to processors in a way that is related to the mapping of another data object. This is useful to ensure that locality in the data structures (arrays) is correctly mapped on the hardware in order to minimize communication.

The result obtained on the IBM SP2 system with HPF is presented on figure 3.6. The results appear to be very poor since we could obtain only a global performance of 0.6 Mflop/s per processor³. However, the performance model still fits perfectly the measurements. As we will see in the next subsection, better performance are obtained using the message passing programming model. In conclusion, nowadays data parallel compilers appear to be relatively immature and the programming model must be chosen as close to the hardware as possible.

3.3.3 Message Passing Programming Model

Message-passing has emerged as one of the more widely-used paradigms for expressing parallel algorithms. Its expressive power has proven adequate, if not always convenient, for writing parallel programs; and its close match with the hardware of many current parallel computers (including workstation networks) has made it possible for the programs to exploit the full capabilities of most current hardware. Although it has shortcomings, it comes closer than any other

³This means that we must theoretically dispose of 27 processors to obtain with HPF the same performance than the sequential code run on one processor!

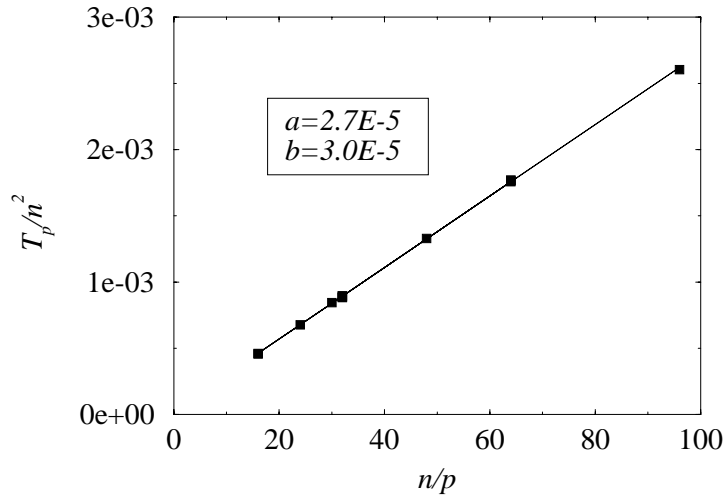


FIGURE 3.6: Performance analysis of the LB Benchmark written in High Performance Fortran. The code was run on a IBM SP2 Power system. The performance model suits perfectly the data. We have used arrays of size ranging from 128 to 512 and up to 12 processors. However the overall performance is strikingly low compared to the same machine with a code written with Message Passing programming model: see figure 3.8

paradigm to being a “standard” approach for implementation of parallel applications.

Programming with message passing is more complicate since the user makes calls to libraries, MPI⁴ or PVM⁵[32] for instance, to explicitly manage information sharing and synchronization between the processes. It is consequently more general and allow the so called Multiple Program Multiple Data (MPMD) approach. The message passing programming model can be considered as a lower level model since most of data parallel compilers actually do use libraries like PVM or MPI to achieve their communication and synchronization.

The most basic type of operation performed by a message libraries are the so called point-to-point communication where a defined processor (the source) sends the message to the another processors (the destination) whereas the destination must perform the matching operation of receiving the message. Typical message passing libraries subdivide the basic communications into two types: blocking - processing waits until message is transmitted and non-blocking - processing continues even if transfer has not completed.

Using message passing for our typical data parallel problem consists of partitioning the arrays into bands or contiguous blocks and assigning them to the set

⁴Message Passing Interface, see the MPI forum documents at <ftp.mpi-forum.org> in [pub/docs/](#)

⁵Parallel Virtual Machines

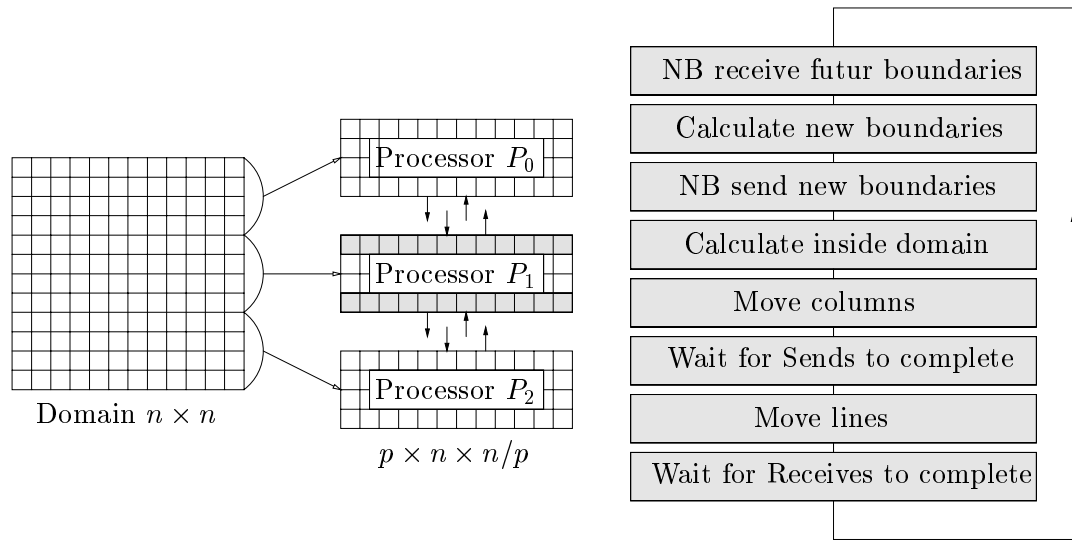


FIGURE 3.7: Schematic representation of the different steps involved in the message passing implementation of our algorithm. The original domain of size $n \times n$ is decomposed in slabs of equal widths and distributed among the p available processors. Each processors is then running the same code which applies itself to best overlapp computation and communication time.

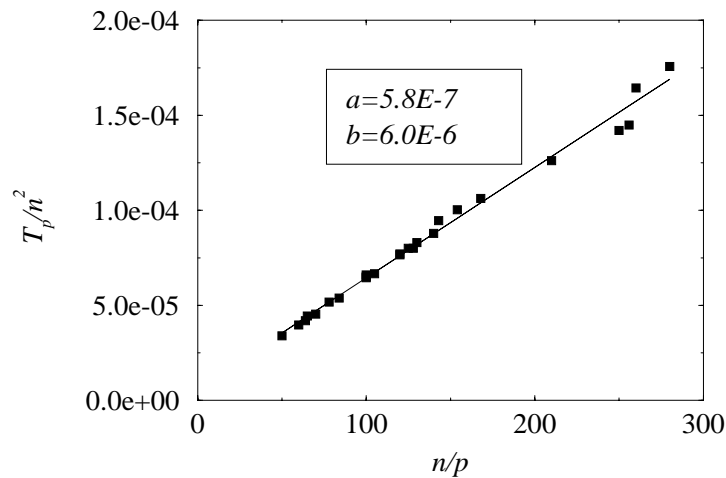


FIGURE 3.8: Performance analysis of the LB benchmark running on an IBM SP2 Power system under MPI. The performance model is well suited for the value n of ranging from 500 to 1000 and we have used up to $p = 14$ processors. The implementation is 50 times faster than with HPF (see figure 3.8)

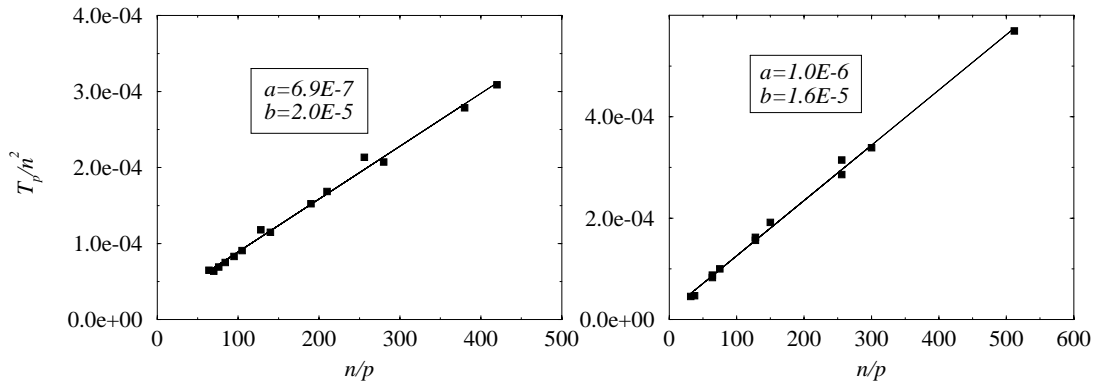


FIGURE 3.9: Performance analysis of the LB benchmark running on a cluster of Intel Pentium Pro 200 (on the left) and on a Cray T3D (on the Right), both under PVM. The model given in 3.1 is well suited to the performance of both machines. Here we have used up to 10 processors for the PC cluster and up to 16 processors on the T3D. The system size are ranging between $n = 500$ and $n = 1000$.

of available processors. No functional decomposition is needed and no master-slave distinction is useful. A band residing on a processor consists of interior and boundary-points. We call boundary points the two strips (see figure 3.7) that have neighbors sitting on another processor. These boundary-points must be transmitted to neighboring processors as the streaming phase takes place. The synchronization is also implicitly embedded in this streaming phase since the update of a site at time t requires the knowledge of the neighboring sites at time $t \pm 1$. Synchronization is achieved by waiting - just before the next iteration is started - until all sends and receives have completed.

Using blocking communication is not efficient because the key point to obtain performance is to minimize overhead and latency by carefully overlapping communication and computation through use of nonblocking (NB) point-to-point communication. Figure 3.7 is collating the “in-slice” data partitioning scheme for system size $n \times n$ on p processors and the pseudo-code of the algorithm run by each processors.

The performance obtained on the IBM SP2 Power system using MPL, an IBM implementation of MPI, is shown on figure 3.8. The performance model is again well suited to the problem and we obtain an overall performance in site update frequency of about 1.5 MHz per processor. Thus, we need at least 12 processors to run our simulation as fast as on the Connexion Machine. We have increased flexibility concerning the choice of problem size at the cost of an explicit parallelization of the communication. Compared to HPF on the same machine we can observe that message passing has improved the performance by a factor of about 50!

One may observe that the load may be better balanced since the processors containing the source is doing more work than the peripheral processors. We could also avoid the communication until there is really something to communicate, namely as long as the fields are empty. This could be achieved by a more complicated and less flexible domain decomposition. We have not implemented such refinements because it would be dependent on the initial condition which may vary a lot from one LB wave application to another. Moreover, dynamical load balancing, namely dynamical partitioning of the domain turns out to be essential for data parallel programs running on heterogeneous parallel machines[33]. But for our case, communication is so important (it is responsible for about 30% of the time spent on the SP2 using a high performance switch and interconnection network) that only dedicated communication network and preferably dedicated machines are efficient.

The extreme portability and the maturity of the PVM [32] library allows us to run our algorithm in parallel on a wide variety of machines ranging from the expensive massively parallel Cray T3D to a cheap but very powerful cluster of Intel Pentium Pro 200. The results on these machines are shown on figure 3.9. The cluster of Pentium Pro 200 gives surprisingly good performances especially if price considerations are taken into account. Indeed, for the relative small sizes considered here, the T3D gives an overall performance which is 30% lower than the PC cluster despite a 100 times higher price.

3.3.4 Shared Memory Programming Model

In the shared memory model, the programmer is giving explicit parallelization directive to the compiler which can produce a code that can run concurrently on multiple processors. Unlike the data parallel model, these directives are not responsible for data partitioning but rather control the succession of parallel and serial segment of the code. Synchronization is achieved by controlling tasks-reading from and -writing to the shared memory.

Conceptually, upon entering a parallel segment, namely a list of instruction that may be performed independantly (and thus parallely) on some array data portions, the application splits into a number of threads which then execute various portions of the parallel segment at the same time. At the end of the parallel segment there exists a barrier at which the threads are collected together until all are finished executing, then the execution of the next serial segment begins. A parallel segment is specified by the instruction or the subroutine following the directive `#pragma parallel`, in C.

In our case, the key point is to parallelize the loop over the domain lines. A unique directive is marking a for-loop for parallelization: `#pragma pfor`, and some more indications must be given to the compiler such as the list of variables that are shared and the list of local variables. As we can see, the shared memory programming model is easily used efficiently on a corresponding hardware. The

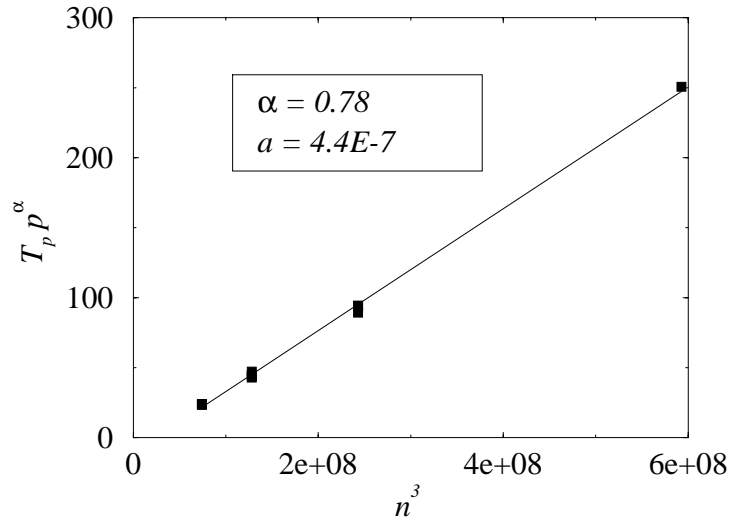


FIGURE 3.10: Performance analysis of the LB benchmark written for the parallelizing compiler on the Origin 200 shared memory machine. The performance model correspond to equation (3.2) and fits well the data corresponding to various simulations running on 1,2,3 and 4 processors. As expected, the slope give a 1-processor performance higher than what was found for the SP2, but an exponent $\alpha = 0.78$ dramatically lowers the contribution of the added processors. We will not give an interpretation of the intercept constant which turns out to be negative here.

coding is simple but it is not portable and often provides little control over interprocessor data transfer costs. Another disadvantage is the lack of scalability that comes from the fact that memory is bandwidth limited and the number of processors may not indefinitely increase without causing severe bottlenecks.

Starting from a sequential version of the code given in appendix B (*i.e.* without the calls to PVM routines and after some cosmetic rewriting), the directives must be inserted to one unique place: at the beginning of the loop coding the collision process. This is done that way

```

/*****                               [...]                               *****/

#pragma parallel
/***** The next program unit, namely the loop over the lines for(i=0;...)*****/
/***** will be executed in parallel                                     *****/

#pragma shared(dim,f,delay,coeff,integral)
/***** The variables specified must be shared among all the processors *****/

#pragma pfor
/***** The parallelism is introduced as a parallel for. The iterations *****/

```

```

/***** of the for(i=0;...) are split in equally weighted parts and      *****/
/***** entrusted to each process                                         *****/

#pragma local(psi,i,j,temp,coef)
/***** The variables specified must be local to each processors          *****/
/***** Each processor has a local copy of those variable                 *****/
    for (i=0;i<dim;i++)
        for (j=0;j<dim;j++)
            if(f.f1[i][j]!=0.0 || f.f2[i][j]!=0.0 ||
                f.f3[i][j]!=0.0 || f.f4[i][j]!=0.0  ){

                f1 = f.f1[i][j];
                f2 = f.f2[i][j];
                f3 = f.f3[i][j];
                f4 = f.f4[i][j];

                delay[i][j]++;
                psi=(f1+f2+f3+f4);

                integral[i][j]+=psi*psi;
                psi*=coeff.c2[i][j];
                coef=coeff.c1[i][j];

                f.f1[i][j]=coef*(psi-f3);
                f.f2[i][j]=coef*(psi-f4);
                f.f3[i][j]=coef*(psi-f1);
                f.f4[i][j]=coef*(psi-f2);
            }
/*****                               [...]                               *****/

```

The performance obtained using a four-processors Origin 200 shared memory machine is shown on figure 3.10. As expected, the data does not fit the performance model (3.1). Strictly speaking, there is no communication - in the sense that there is no exchange of boundary - during the execution time. But this does not mean that we must observe a perfect linear speedup. Qualitatively, we observe, for instance, that a single processor of the Origin 200 is faster than a SP2 thin node by a factor of 1.5. A two-processor run gives a significant speedup, yet smaller than 2. The gain of adding more processors is low. Used with 4 processors, the Origin 200 is slower than 4 SP2 nodes.

The goal is again to provide a performance analysis of our algorithm which is simple and independant of the problem size $n \times n$ and the number of processors p involved. We propose to analyse the execution time of n iterations T_p according to the following new, empirically deduced, scaling law

$$\frac{T_p}{n} = a \frac{n^2}{p^\alpha} \quad (3.2)$$

For simplicity, the execution times was the wall-clock time as no other users were allowed during the performance measurements. As in equation (3.1) a represent

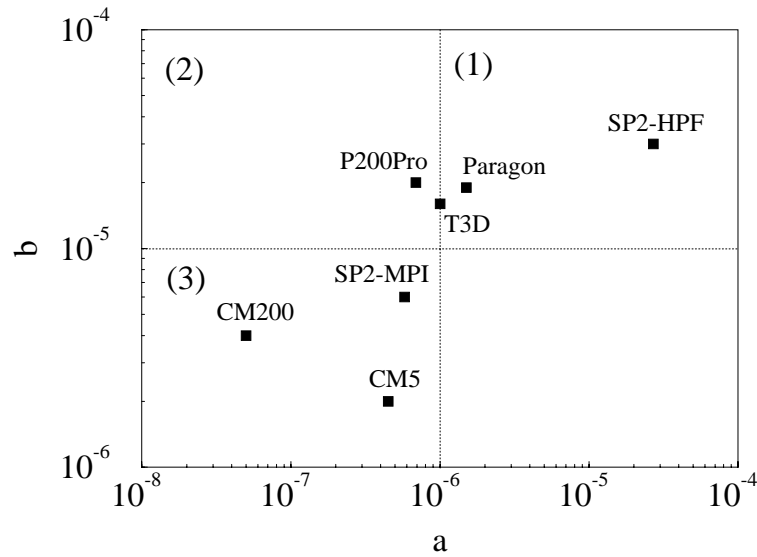


FIGURE 3.11: This figure is a “performance map” for parallel machines based on the performance model (3.1). The cities represent the machines tested throughout chapter 3 plus two other machines; see [34]. The regions of the map indicate (1) slow computation, slow communication, (2) fast computation, slow communication and (3) fast communication, fast computation. This comparison is independent of the number of processors and the size of the domain.

the time unit of a site calculation so that $14/a$ is the effective number of Flop per second for one processor. The exponent $\alpha < 1$ lowers the efficiency of additional processors. It accounts for the time spent during the synchronization of the threads, the management of the various barrier and the migration of the data in the existing memory hierarchy. The reader may wonder if so much spare time may be reduced to a single exponent ! On figure 3.10 the data fit well the performance model and we obtain an exponent $\alpha = 0.78$ which is a first insight in the overall quality of the shared memory architecture. We consider this model as a promising approach of shared memory machine but it should confronted different machines to gain further credibility.

3.4 Scalability

One advantage of the performance model given by equation (3.1) is that it allows us to calculate straightforwardly the speedup and the scalability. For the implementation fitting the model (3.1) we may write the speedup as follows:

$$\text{Speedup} = \frac{T_{\text{sequential}}}{T_{\text{parallel}}} = \frac{an^2}{a\frac{n^2}{p} + bn} = p \frac{1}{1 + \frac{bp}{an}}$$

This means that our numerical model is linearly scalable on the wide variety of parallel machines and programming models we have studied. Unfortunately, it is not the case for the performance model (3.2). The efficacy of the shared memory machine, tested with our program, decreases with an increasing number of processors, whatever the size of the problem.

Finally, figure 3.11 summarizes the performances measured throughout this chapter and the performances presented in [34]. By collecting the various coefficients a , b defined in (3.1), we may draw a map with different regions characterizing the speed of the computation and the communication.

Simulation of Radio Waves Propagation

The fast development and the growing importance of mobile and personal communication systems such as cellular phones demands increasingly better planning tools to ease the deployment of wireless communication networks. An efficient planning is based on accurate predictions of radio-wave propagation and, in particular, a detailed knowledge of wave propagation in heterogeneous media such as urban environment is required.

Urban environments are of strong interest as they concentrate a large number of users. However, they constitute a difficult wave propagation problem which is studied by various authors[35, 36, 37]. Radio waves are absorbed, reflected, diffracted and scattered in a complicated way on the buildings. In addition one cannot exclude that the signal is partially transmitted through a building or over its roof. Therefore, the amplitude pattern of a wave emitted by an antenna surrounded by such obstacles is quite complicated and beyond an analytical calculation.

Yet, the coverage region of an antenna is a crucial question in the development of the future mobile communication systems in a city. As the number of users will increase, a distributed network of base stations will be necessary in order to accommodate for the growing traffic. Each of these base stations will be in charge of a given region (a cell), whose size is defined so that the expected number of calls within this region matches the limited number of channels available at the station. When a user reaches the boundary of the region served by a given station, a nearby base station must take over the transmission and commute the call to a new radio resource (usually a different frequency channel). Therefore, adjacent cells must overlap.

The planning and maintenance of a mobile communication network is a hard optimization problem. The base stations must be placed in appropriate locations so that a complete coverage is guaranteed with a minimum number of cells, each of them no larger than what is allowed by traffic or propagation requirements. In addition, a disjoint set of radio resources must be assigned to nearest neighbors cells. Finding such an antenna layout was addressed *inter alia*, using our method and the so-called Genetic Algorithm (GA) technique; see [38, 39] for more details.

Our approach to compute radio wave propagation in a urban cell is based on the Lattice Boltzmann method presented in chapter 2. We restrict ourselves to scalar waves, which turns out to be sufficient to capture the essential behavior. Our time-dependent simulations can be performed with a wave length appropriate to the spatial discretisation providing a renormalization to obtain path loss predictions corresponding to a much shorter wave length used in real measurements. We show that this method is well suited to the present problem and offers promising prospects of development. A software called *Parflow* was designed and developed for the swiss Telecom PTT¹ and this section will go through the details of the features it includes (see appendix C to understand how the present implementation actually works). It offers an interesting alternative to the ray tracing method commonly used in the radio wave propagation community. Finally, the kernel of our approach is one of the corner stones of the European project STORMS[40, 41] developed, among others, at the Swiss Institute of Technology, EPFL[42].

4.1 The numerical model

Our numerical model for simulating wave propagation corresponds to the Lattice Boltzmann approach without rest particles as described in the chapter 2. More precisely, we have used the model without rest particles corresponding to the matrix given in (2.52) The space is discretized as a finite regular square lattice of dimension $N_x \times N_y$ each cell representing a square portion of the real space of dimension $\lambda^2[m^2]$. This cell may eventually represent free space, or a building part, or any other component of a 2-dimensional projection of the real layout. The time increment of the simulation τ is fixed by the speed of the simulated wave c :

$$c = \frac{\lambda}{\tau\sqrt{2}} = c_0 \approx 3 \cdot 10^8 \text{ m} \cdot \text{s}^{-1}$$

The local value of the radio signal, say one spatial component of the electric field for instance, is given by the scalar

$$\psi(t) \doteq f_1(t) + f_2(t) + f_3(t) + f_4(t)$$

¹At the publication date of this dissertation, the company's name change to SWISSCOM

Where the f 's are 4 real-valued variables defined over all the discretized space. We will work with harmonic fields of a given frequency but our simulation takes place in the time domain; thus the instant value ψ is of less importance: it is the amplitude A and possibly the phase θ of the stationary signal that is relevant for our predictions. These quantities are easily obtained by integrations of the signal over one or, if we want to obtain a time averaging, more periods. We have the following definitions:

$$A(\mathbf{r}, t) \doteq \sqrt{\int_{t-2T}^t \frac{\psi^2(\mathbf{r}, u)}{T} du} \quad (4.1)$$

$$\theta(\mathbf{r}, t) \doteq \begin{cases} \arcsin \left(A(\mathbf{r})^{-1} T^{-1} \int_{t-2T}^t \psi(\mathbf{r}, u) \cos\left(\frac{2\pi}{T}u\right) du \right) & \text{if } F(\mathbf{r}, t) > 0 \\ \pi \Leftrightarrow \arcsin \left(A(\mathbf{r})^{-1} T^{-1} \int_{t-2T}^t \psi(\mathbf{r}, u) \cos\left(\frac{2\pi}{T}u\right) du \right) & \text{otherwise.} \end{cases} \quad (4.2)$$

where

$$F(\mathbf{r}, t) \doteq \int_{t-2T}^t \psi(\mathbf{r}, u) \sin\left(\frac{2\pi}{T}u\right) du \quad (4.3)$$

where T is the discrete, integer, period of the signal. Indeed, as we shall see later on, there is a source or an antenna somewhere in the domain, which radiates a sine signal of period T . If the amplitude $A = 0$ the phase $\theta \doteq 0$ by convention. In the radio community the amplitude of a signal is often given by the path loss P expressed in dB.

$$P = 20 \log \left(\frac{A}{A_0} \right)$$

where A_0 is a reference amplitude expressing the power of the transmitter. The evolution of the f 's is given, in the vacuum, by the regular wave LB dynamics. However, special care must be taken to account for the various boundary conditions of the present problem such as the source, the buildings and the boundaries of the domain. Consequently we write the dynamics as follows:

$$f_i(\mathbf{r} + \mathbf{v}_i, t + 1) = c_1(\mathbf{r}) \left(c_2(\mathbf{r}) \psi(\mathbf{r}, t) \Leftrightarrow f_{i+2}(\mathbf{r}, t) \right) \quad (4.4)$$

where \mathbf{v}_i accounts, as usual, for the four main lattice directions. The values c_1 and c_2 are the local factors that determine the nature of the site: vacuum, transmitter, building's wall or system boundary. For the vacuum we have $c_1 = 1$ and $c_2 = 1/2$ and we meet again the evolution given in 2.52. The management of various boundary conditions by simply adjusting local parameters of propagation will be presented in the following subsections. This constitutes the essential advantage of the LB method compared to other method like ray-tracing: all informations are microscopically distributed among the array and only nearest neighbors communication are involved.

4.1.1 Point Sources

We define a sinusoidal source node on the lattice (at the position \mathbf{r}_0 corresponding to the transmitter location) as a site where all the f 's take the same value:

$$f_i(\mathbf{r}_0 + \vec{v}_i, t + 1) = \gamma \sin(2\pi t/T),$$

where γ is a normalization factor which depends on T , the imposed period of the signal. The value of γ is appropriately chosen so that the surrounding vacuum space “sees” a unit-amplitude source, e.g.

$$\psi(|\mathbf{r} \leftrightarrow \mathbf{r}_0|) = 1/(2\sqrt{4\pi|\mathbf{r} \leftrightarrow \mathbf{r}_0|/\mathcal{L}})$$

where \mathcal{L} is the simulation wave length. Determining the appropriate value of $\gamma(T)$ is not straightforward. Indeed, due to the anisotropy of the dispersion relation, we already know that our simulated ψ is, in free space, not only a function of $|\mathbf{r} \leftrightarrow \mathbf{r}_0|$ but also a function of ϕ the angle between $\mathbf{r} \leftrightarrow \mathbf{r}_0$ and a lattice direction. So, suppose we have a unit amplitude source in free space, the resulting $\psi_{\mathbb{I}}$ may be written as

$$\psi_{\mathbb{I}}(\mathbf{r}, \phi, T) = \frac{\psi(\phi, T)}{4\gamma(T)\sqrt{\pi\sqrt{2}|\mathbf{r} \leftrightarrow \mathbf{r}_0|/T}} \quad (4.5)$$

Equation (4.5) is the actual definition of $\psi_{\mathbb{I}}$, the anisotropic correction due to the lattice. The value of $\psi_{\mathbb{I}}/\gamma$ may be numerically determined and is shown on the left hand side of figure 4.1 for the case $T = 8$. Because the dispersion relation is correct on the diagonal *i.e.* for $\phi = \pi/4$, we define γ by imposing $\psi_{\mathbb{I}}(\phi = \pi/4, T) = 1$ for all possible T (*i.e.* the propagation is considered correct on the diagonal). So, finally, γ is defined by

$$\gamma(T) = \frac{1}{\psi_{\mathbb{I}}(\mathbf{r}, \phi, T)4\sqrt{\pi\sqrt{2}|\mathbf{r} \leftrightarrow \mathbf{r}_0|/T}} \Big|_{\phi=\pi/4}$$

The right-hand side of figure 4.1 gives the values of γ as a function of the period of the source. The equation above means not that γ is a function of $|\mathbf{r}|$: we get the same value for different everywhere except in the vicinity of the source. We can observe that γ is an increasing function of T , this is easy to interpret since the power must remain the same but the effective “surface” occupied by the source (*i.e.* 1 pixel) scales like $1/T^2$ if measured in units of the actual wave length.

Due to the discrete nature of the dynamics, it is necessary to have a large enough T to screen off the lattice anisotropy of the dispersion relation. If not otherwise specified we choose $T = 8$ as the smallest acceptable value. Consequently, our generic source has the frequency given by

$$\nu = \frac{c_0\sqrt{2}}{T\lambda} \text{Hz}$$

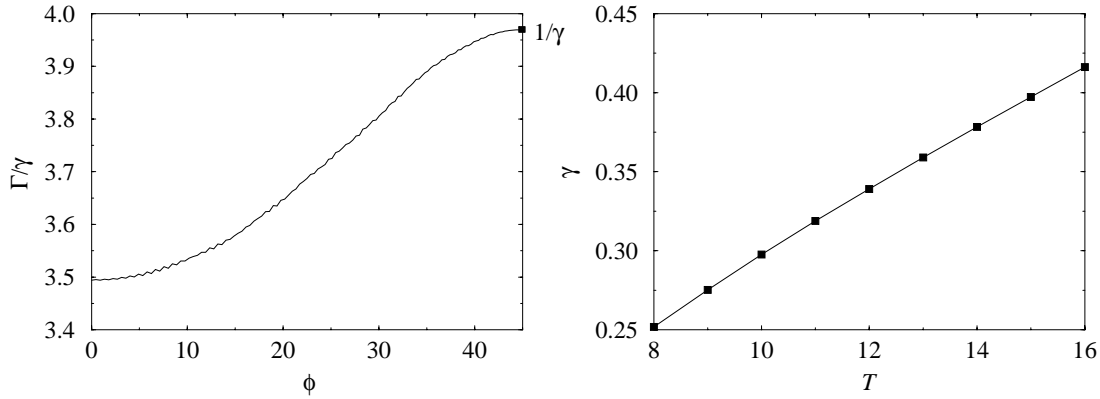


FIGURE 4.1: Left: Effect of the anisotropic dispersion relation for the case $T = 8$; the value of $(T, \phi)/\gamma(T)$ is shown for a unit source. Right: the different values of γ as a function of the period T of the source. A linear dependence is suggested by the solid line.

A typical experiment starts in a discretized space initialized with 0 valued f 's when the source is switched on. The f 's will propagate throughout the sample and eventually reach the boundary of the system. It is interesting to note that everywhere except on the source position, the f 's of a given site have different phases and amplitudes; see figure 4.2 for an illustration. This is an indication of the richness included in the f 's: they cannot be reduced to a single value (like in the discretized wave equation model) without losing their vector contents (*c.f.* the current \vec{J} of the theory given in chapter 2) related to the wave vector \mathbf{k} .

4.1.2 Sites of Reflection

A site of reflection is defined on every wall location, for instance. It returns all incoming flux with opposite sign and direction. An attenuation factor $\mu \in [0, 1]$ is heuristically fixed in order to account for the reflection coefficient relating incident to reflected energy at the considered boundary. The evolution of the f 's is then given by

$$f_i(\mathbf{r}_0 + \vec{v}_i, t + 1) = \Leftrightarrow \mu f_{i+2}(\mathbf{r}_0, t) \quad (4.6)$$

where $i + 2$ is wrapped on 1, 2, 3, 4. Sites of reflection corresponds to the choice $c_1 = \mu$, $c_2 = 0$ in equation 4.4. The coefficient μ is not straightforwardly related to the desired reflection coefficient. In particular, we will see later that putting $\mu = 0$ is not an efficient way to simulate perfect absorption. μ is the most important parameter to adjust when prediction has to be made on a particular city layout: it takes values typically around $0.6 \Leftrightarrow 0.7$, but must be adjusted to conform to available measurements, if any. An interesting property of the model

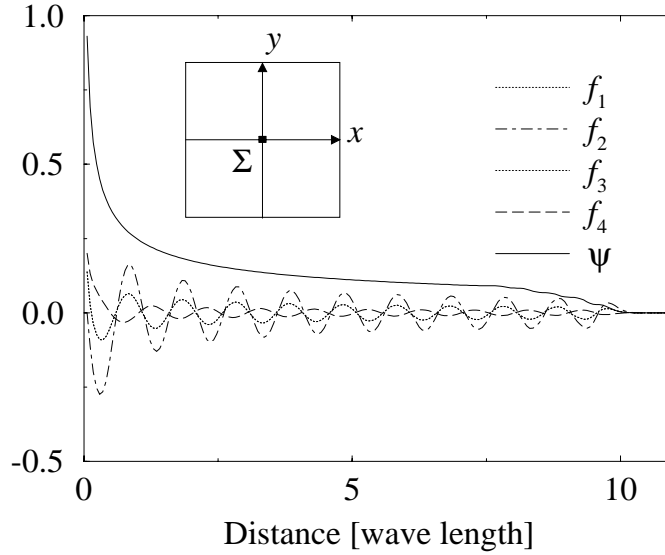


FIGURE 4.2: The different f 's and ψ are shown along the positive y axis; $\gamma = 1$. Layout and position of the source Σ is shown in the inset. We can observe that the value of ψ behaves well even in the proximity of the source, e.g. every points may treated as “far” from the source. There is a front zone of 2 wave lengths in which the ψ has not reached its stationary value. f_4 is out of phase; this is because f_4 is traveling backwards along this axis. f_1 and f_3 are the same and take a value which is half of f_2 . The wave vector is parallel to $(f_1 \Leftrightarrow f_3; f_2 \Leftrightarrow f_4)$.

is its capability to choose different coefficients for each building or even built up walls consisting of different coefficients without any additional computations. However, those data are simply not yet available and that's why we usually have chosen uniform μ while simulating a real urban area. The figure 4.3 shows the value of the effective reflection coefficient effective reflection coefficient resulting for different values of μ and for different angles of incidence. The particularity of the angle of incidence 45 deg is particularly well visible for $\mu = 0$

4.1.3 Sites of attenuation

Sites of attenuation corresponds to regular vacuum sites multiplied by a damping factor μ . This corresponds to the choice $c_1 = \mu$, $c_2 = 1/2$ in equation 4.4. The difference between sites of reflection and sites of attenuation is when closed paths are made of these sites, corresponding to buildings wall for instance. A reflecting boundary is not permeable and no waves are allowed to go through the building whereas an absorbing boundary do not prevent penetration of the waves but still produced some reflected waves. This point will be exploited later on in this section.

Sites of attenuation constitute the essential idea of a crucial point, namely

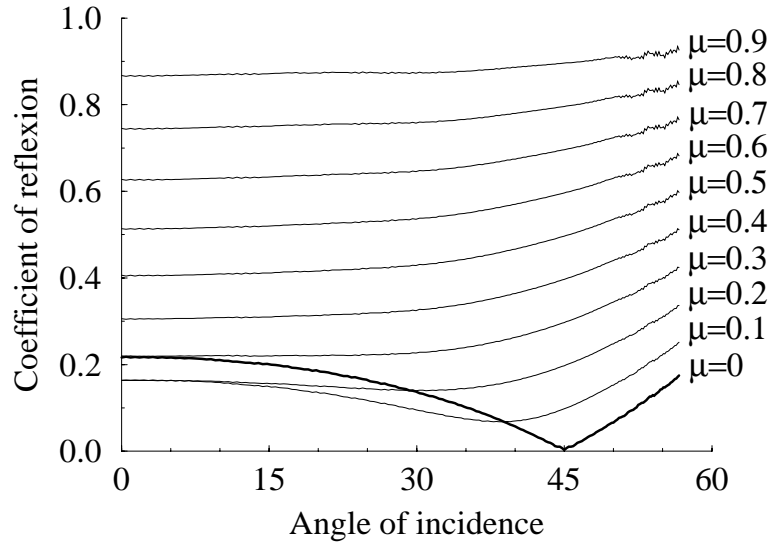
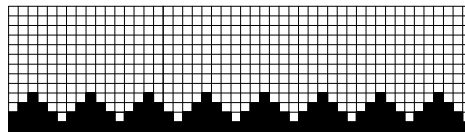


FIGURE 4.3: *Effective reflection coefficient: $A_{\text{ref}}/A_{\text{inc}}$ relating reflected and incident wave intensity in function of the coefficient μ and the angle of incidence. The layout of the experiment is following: a source is put in the center of the system and a “wall” of reflecting nodes parallel to the y axis, is put between the source and the system boundary.*

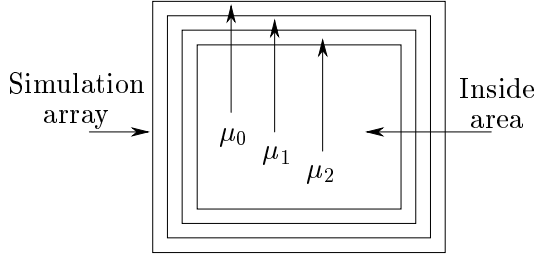
the built up of nearly perfect absorbing boundary framing our system. As a matter of fact, to simulate infinite space on a finite lattice we must gradually absorb the energy near the boundary, limiting as much as possible the reflection due to the breaking of the spatial homogeneities. This problem is very general in the simulation community and the way it is solved, often determine the issue of the whole modelisation. The ease with which perfect boundary are built within our approach is an important advantage of our method.

As we have already seen in figure 4.3 setting all the outgoing f 's to 0 on the boundary is far from being efficient as about 20% the normal incident intensity is reflected. However, because the adsorption is perfect for an incident angle of 45 degrees, it is possible, by unsmoothing the boundary the way shown in the figure below, to reduce this reflection down to 7% of the incident intensity.



By the way, the shape proposed here looks similar to the shape of common acoustic isolation walls. But the best way remains to frame the domain by three (or possibly more) layers of sites of absorption. The coefficient is decreasing from the inside of the domain as shown below and depends on the discrete period involved. The best coefficients found for different values of T are listed in the

table hereafter.



T	μ_2	μ_1	μ_0	A_r/A_i
8	0.8	0.436	0.0	$6.1 \cdot 10^{-3}$
10	0.89	0.57	0.0	$3.7 \cdot 10^{-3}$
12	0.83	0.55	0.0	$3.3 \cdot 10^{-3}$

There is another important application of the sites of absorption, namely the pseudo 3D building layout simulation. The introduction of sites of absorption on building wall allows part of the waves to penetrate through the buildings. This may be a valuable technique which can take into account propagation over buildings roofs despite the two-dimensional aspect of our model. This technique was particularly valuable (see [43]) when the layout is discretized such that streets begins to be excessively narrow compared to the simulation wave length, thus artificially preventing wave penetration. No general effectiveness and accuracy of this method is given here because it is dependent of the details of the building layout. A more subtle use of this method should be further investigated; for instance the buildings far from the source could be considered as transparent whereas the buildings near the source are rather “opaque” e.g. made up of sites of reflection.

4.2 The Renormalization Method

When the transmitting antenna is below the rooftop, the essential features of a building layout is assumed to be captured by a two-dimensional simulation (which is equivalent to a line source and infinitely high buildings). But some kind of renormalization must be taken into account when considering the propagation of a point source in a three-dimensional space. In fact, we have the following relation between the two- and three-dimensional free space propagation amplitude A^0 :

$$A_{2D}^0(\mathbf{r}) = \frac{1}{2\sqrt{4\pi d(\mathbf{r})/\mathcal{L}_0}}$$

$$A_{3D}^0(\mathbf{r}) = \frac{1}{4\pi d(\mathbf{r})/\mathcal{L}_0} = \frac{2A_{2D}^0(\mathbf{r})}{\sqrt{4\pi d(\mathbf{r})/\mathcal{L}_0}}$$

where $d(\mathbf{r}) = |\mathbf{r} \leftrightarrow \mathbf{r}_0|$ is the distance to the source which is supposed to be much bigger than \mathcal{L}_0 its wavelength. Similarly, for an inhomogeneous medium like an urban microcell, we assume that the pattern obtained with our 2D simulation

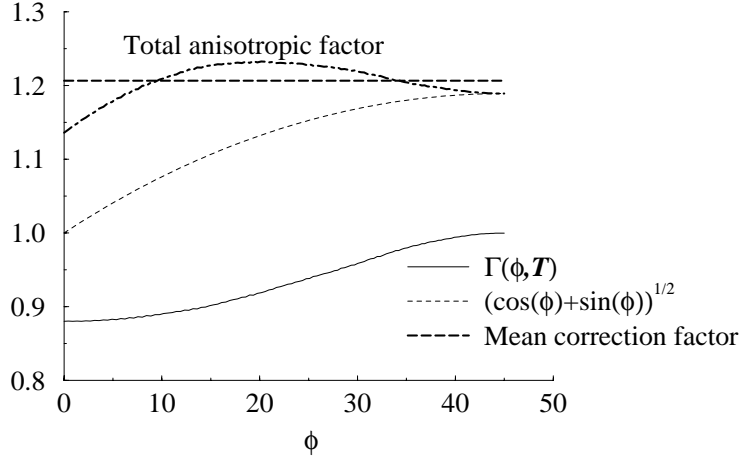


FIGURE 4.4: Plot of the , factor and the correction due to the anisotropy of the Manhattan distance for the case $T = 8$. The total correction factor defined in 4.7 is shown as well as the mean correction factor which can be absorbed in γ .

If the Manhattan distance is used, some special consideration may be added. The Manhattan distance is exact along axis directions but overestimates the diagonal distances by a factor of $\sqrt{2}$. This anisotropy may be used to slightly thwart the effect of the anisotropy of the dispersion relation which is exact precisely on the diagonals and shorten the wavelength along axis direction. Recalling the definition (4.5) of , we can write the error of our renormalization, in case of the free space propagation, focusing on the anisotropic contributions of both , and the Manhattan distance:

$$\begin{aligned}
 A_{3D}^{\text{real}} &= A_{2D}^{\text{cont}} \frac{\mathcal{L}_0}{\sqrt{\pi \mathcal{L} d'_{\text{cont}}}} \\
 &= A_{2D}^{\text{sim}} \frac{\sqrt{\cos(\phi) + \sin(\phi)}}{, (\phi, T)} \frac{\mathcal{L}_0}{\sqrt{\pi \mathcal{L} d'_{\text{Man}}}} \quad (4.7)
 \end{aligned}$$

where A_{2D}^{cont} is the continuum limit of A_{2D}^{sim} and d'_{cont} the correct isotropic distance. The factor $\sqrt{\cos(\phi) + \sin(\phi)}$ is the correction introduced by the change from the real distance d'_{cont} to the Manhattan distance d'_{Man} : ϕ is the angle between $\vec{r}_{\text{observation}} \Leftrightarrow \vec{r}_{\text{source}}$ and a main lattice direction. The term $1/$, accounts for the the anisotropic error due to the lattice contribution; see (4.5). It is plotted on figure 4.4 for the case $T = 8$. Its average contribution with respect to ϕ may be corrected by changing the value of γ :

$$\gamma \rightarrow \frac{\pi \gamma}{4 \int_0^{\pi/4} d\phi \frac{\sqrt{\cos(\phi) + \sin(\phi)}}{\Gamma(\Phi, T)}}$$

Finally, for $T = 8$ (the value retained for most of our simulations) and renormalizing with the Manhattan distance we have to choose $\gamma = 0.2519 \times 1.2067 = 0.304$:

$$\boxed{\gamma_{(T=8)} = 0.304}$$

4.3 The Simulated Wave Length “Problem”

The distinction between \mathcal{L} and \mathcal{L}_0 , introduced in the previous section was not innocent. In fact, there is an intrinsic limitation on the simulation wave length as soon as comparisons with real measurement is performed. Although computer memory increases every day, this is due to the limited memory currently available. We find it reasonable to work with lattice size smaller than one million sites (1000×1000). Usually the urban cells under investigation have an extent of few hundreds meters, therefore $\lambda \approx 10^{-1}$. On the other hand the smallest possible period on the lattice is $T_{\min} = 2\sqrt{2}$, but we found that for $T \leq 6$ the model fails to simulate accurate propagation and the dispersion relation becomes too bad; see figure 2.5 for instance. Consequently, typically $T \approx 10$ which yield a limited simulation frequency given by

$$\nu = 400\text{MHz}$$

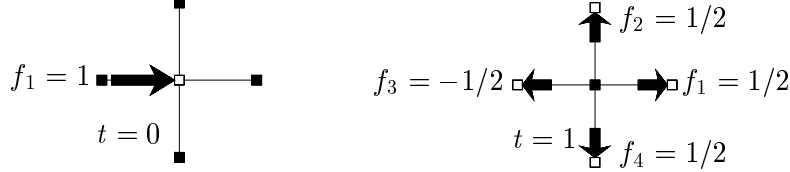
This frequency turns out to be much lower than the spectrum of frequencies used in the radio wave propagation community for the mobile communication system which are around 1800 Mhz.

This is not as dramatic as it appears at a first glance; our simulations are performed at a wave length which is appropriate to the computational requirement. It is much larger than the actual wave length used in real measurements but we show how to account for this problem and correct the pathloss given by our numerical predictions using the renormalization technique. Simulating a high and correct frequency signal does not really make sense because it would require to have a precise knowledge of the small scale obstacles (of dimension down to 10 cm) that would affect the propagation at this short wave length. Clearly, the building roughness and the presence or absence of cars in the street or even opened windows are information which is simply not available. For this reason, a simulation at the correct wave length is out of reach for present time models and our results are to be taken as an average intensity over a spatial region of a size comparable to the wave length we use. The good agreement with the real measurements presented in section 4.6 indicates that our procedure is valid, at least for the approximation level which is necessary in this problem and that the effect of the disorder is well taken into account in our approach.

On the other hand, we cannot chose an arbitrarily large wave length. The simulation wave length must be shorter than the dimension of the relevant large scales inhomogeneities, namely the width of the streets.

4.4 Huygens Principle

In optics, the principle of Huygens (in *Traité de la lumière*, 1690) is a statement that all points of a wave front of light in a vacuum or transparent medium may be regarded as new sources of wavelets that expand in every direction at a rate depending on their velocities. It is a powerful method for studying various optical phenomena and our LB wave model can be interpreted as a discrete and microscopic formulation of this principle, as can be seen on the following figure where the evolution of a 1-valued f_1 is shown:



The principle of Huygens may be strictly formulated; the value of the wave amplitude A anywhere in a closed volume \mathcal{V} containing no source, may be expressed by the value of A and ∇A taken on the boundary $\partial\mathcal{V}$ of the volume:

$$A(\vec{a}) = \int_{\partial\mathcal{V}} d\vec{s} \left(G(r) \nabla A(\vec{x}) \Leftrightarrow \nabla G(r) A(\vec{x}) \right) \quad (4.8)$$

where

$$G(r) = \frac{1}{4\pi} \frac{e^{ikr}}{r}, \quad r = |\vec{x} \Leftrightarrow \vec{a}|$$

is the Green's function of the differential operator $\nabla^2 + k^2$:

$$\Leftrightarrow \nabla^2 G(r) \Leftrightarrow k^2 G(r) = \delta(r).$$

Projecting the gradients of 4.8 on the surface normal \vec{n} we obtain, after some algebra, the Kirschoff integral expression, see [44]:

$$A(\vec{a}) = \int_{\partial\mathcal{V}} \frac{ds}{4\pi} \frac{e^{ikr}}{r} \left(\vec{n} \nabla A(\vec{x}) \Leftrightarrow ik \underbrace{\left(1 + \frac{i}{kr} \right)}_{\rightarrow 0} A(\vec{x}) \vec{n} \vec{e}_r \right)$$

where the analogy with Huygens principle appears clearly: A is built up from a superposition of spherical wavelets. The amplitude of each wavelet source sitting on a closed surface depends upon the local value of A , ∇A and the orientation of the surface normal.

The Huygens principle may be used at a larger scale in our model to introduce new boundaries in our system. The idea is to replace the source by an ensemble of sites continuously embedding the source (Huygens nodes) so that the propagation is not altered outside this envelope; and so, the inside sites may be switched off or forgotten in subsequent evolution. More precisely, the evolution is calculated in two phases:

1. The first phase takes place as usual until a specified area \mathcal{V} around the source location has reached stationary f 's. Then, during two more periods of time, the phase and the amplitude are calculated on the ensemble of Huygens sites using formula (4.1) and (4.2) (the boundary of \mathcal{V}) for each of the f , e.g. not only for ψ .
2. The calculated amplitudes and phases permit to impose on the Huygens sites the asymptotic values reached at the end of phase 1 and thus to continue the propagation without the inside of \mathcal{V} .

In the current implementation of *Parflow* the above described Kirschhoff method for dynamic boundaries generation is an experimental mode. However the method is an interesting alternative to screen off the effect of the system boundary when the source is far from the center of the domain. Moreover, this is also the correct way to introduce new boundaries each time we would like to deallocate some simulation area; this is useful to reduce memory and CPU but yield non rectangular data structures.

4.5 Optimizing the Algorithm

The most important limitation when implementing our model concerns the memory requirement which increases with the square dimension of the urban cell studied. To give an order of magnitude, a simulation performed on arrays 1000×1000 needs about 37 Mbytes memory. The basic idea is to calculate on a restricted domain and thus save memory and also calculation time. In this section we will present two sub-domains decomposition: the static sector-shaped decomposition and the dynamic ring-shaped decomposition. They will take advantage of our efficient, while imperfect, absorbing boundaries framing the computational area and of the fact that the sites near the source reach stationary values much before the periferical points.

To better quantify the gain in efficiency of an optimization method we must fix the approximate number of iterations that should be done on a system of size $d \times d$. The idea is, on the one hand, to wait enough time to allow the waves to reach the boundaries and, on the other hand, to stop before these boundaries (which are non perfect absorbers) alter to much the results. If the source is placed at the center, a good compromise is clearly to choose d iterations, *i.e.* as much iterations as the linear dimension of the domain and it is our choice if not otherwise stated.

4.5.1 The Sector-shaped Decomposition Model

The idea is to split the rectangular domain, supposing the source is at the center, in four smaller, equal and overlapping rectangles. The calculation is then se-

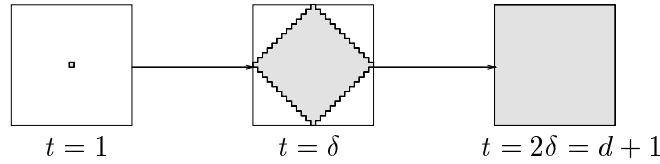
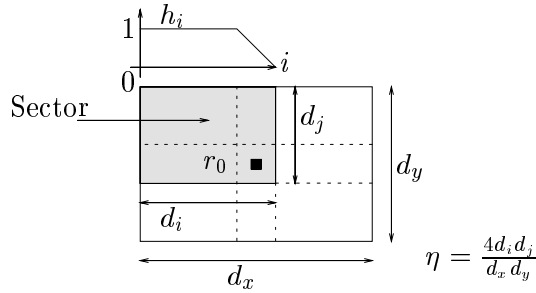


FIGURE 4.5: *Dynamical causality sub-domain of computation (shaded) at various propagation time. The source is in the center and the total number of site updates is of the order $4\delta^3$.*

quentially performed in each of these sub-domains and the different results, e.g. the intensity maps, are finally merged together. The efficiency of the method, in terms of memory saved, and the errors it introduces depends on the overlapping ratio $1 < \eta < 4$:



Indeed, as $\eta \rightarrow 1$ the source is drawn up to the subsystem boundary and thus introduces supplementary errors. The recomposition process consists of the summation of the four sub-domain results with a linear transition factor h_{ij} . For instance, for the domain shaded in the above figure the local transition factor is (Note that the indices i, j are counted from the upper left corner):

$$h_{ij} = h_i h_j, \quad \text{where} \quad h_i = \begin{cases} 1 \Leftrightarrow \frac{i-d_i+2r_{0i}}{2r_{0i}} & \text{if } d_i \Leftrightarrow 2r_{0i} \leq i \leq d_i, \\ 1 & \text{if } i < d_i \Leftrightarrow 2r_{0i}, \text{ and} \\ 0 & \text{if } i > d_i. \end{cases}$$

The technics works good and as been implemented in *Parflow*; see [45], for instance.

4.5.2 The Ring-shaped Decomposition Model

As the waves are propagating in the time domain, it is natural to avoid calculation on the sites outside the causality disc, namely where field values are still 0. In the vacuum, supposing again the source is centered, the causality disc is $r(i, j)$ so that $|i \Leftrightarrow d/2| + |j \Leftrightarrow d/2| \leq t$, where $d = d_i = d_j$ for simplicity. The question is now to determine the number of site updates after d iterations using this method, *i.e.* with a domain growing as shown figure 4.5.

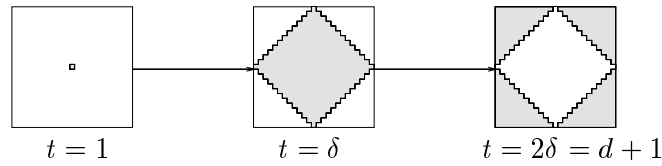
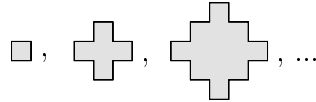


FIGURE 4.6: *Dynamic supersonic ring-shaped domain of computation (shaded) at various iteration time. Here we have computed on a growing ring-like domain with $l = \delta$: the number of site updates is of the order $10/3\delta^3$.*

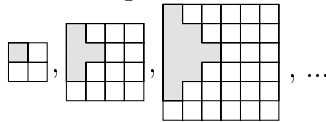
We write $v(i)$ the total number of site updates after i iterations if the domain is continuously growing every iteration as shown here below



which corresponds to the first phase of the calculation *i.e.* for $i \leq \delta = (d + 1)/2$. We have after some algebra

$$v(i) = \sum_{k=1}^i \left((2k \Leftrightarrow 1)^2 \Leftrightarrow 4 \sum_{s=0}^{k-1} s \right) = \frac{2i^3 + i}{3}.$$

But the causality sub-domain is not growing the same way once the boundaries are reached, namely for $i > \delta$. Note that for $i > 2\delta$ the domain is not growing at all anymore; this is the moment we have chosen to stop the simulation. For $\delta < i \leq 2\delta$ however, the number of site updates is less than $v(i)$: $v(i) \Leftrightarrow w(i \Leftrightarrow \delta)$, where $w(j)$ is the excess of site updates due to the fact that the domain considered for $v(i)$ has become bigger than the system limits. It is easy to see that this excess increases every iteration like following:



So, we have after some algebra

$$w(j) = \sum_{k=1}^j 4k^2 = \frac{4j^3 + 6j^2 + 2j}{3}$$

In conclusion, for d (d is odd) iterations in a system of size $d \times d$ using the causality method shown in figure 4.5, we have to update

$$v(d) \Leftrightarrow w\left(\frac{d \Leftrightarrow 1}{2}\right) \approx d^3/2$$

sites instead of d^3 : this is a gain of 2 for typical d which are big enough to neglect terms of order d and d^2 .

Another optimization idea is to stop the calculation on the sites which have reached a stationary value. This procedure introduces new (dynamic) boundaries and thus may dramatically alter the result if it is not done with enough care. The simplest way is to fix *a priori* the upper limit l of iterations that will be performed on each site and to retain the value of the field just before the site is turned off. Thus, the calculation area is a ring-shaped domain with a growing radius; the boundaries are moving with the information velocity. This implies that no error is introduced in the calculation but, the velocity of the waves being smaller than the velocity of information by a factor of $1/\sqrt{2}$, the supersonic moving ring-shaped domain is finally catching the initial front at the iteration time $l/(1 \Leftrightarrow 1/\sqrt{2}) \approx 3.4l$. This may be dangerous if l is too small (but if l is too big, the method is not an improvement!) because the first wave front - which propagates in the virgin domain (zero-valued initialized arrays) - turns out to be lacking in precision. As for the zone before the front, the fields are ... 0, and we eventually measure just those empty sites. Figure 4.6 shows a typical situation where l is acceptable: $l = (d + 1)/2$.

For d iterations in a system of size $d \times d$ using this method, we have to update

$$v(d + 1) \Leftrightarrow w\left(\frac{d + 1}{2}\right) \Leftrightarrow v\left(\frac{d + 1}{2}\right) \approx \frac{5}{12}d^3$$

sites instead of d^3 (the third term accounts for turned off sites): this is a gain of 2.4 for typical d which are big enough to neglect terms of order d^{-1} and d^{-2} . Note that the supersonic method is only 20% faster than the causality method (without introducing too much errors) but only 75% of the memory is enough if a dynamic mapping of sites is implemented. *Parflow* uses the causality method for simplicity.

To avoid the problem of the supersonic moving boundaries catching the front, boundaries have to move with the wave velocity. This has to be done with very much care since these boundaries may affect dramatically the results. The introduction of Huygens nodes, presented in next section 4.4 is the correct way to introduce non-perturbing boundary once the stationary values of the fields are known on an ensemble of sites, but this considerably complicates the numerical implementation.

4.6 Simulations: Calibration & Predictions

Our simulation procedure typically starts with the discretizing of a building layout. This can be easily achieved by scanning a map of the urban area under investigation and directly interpreting the color edges of the resulting image to find out the buildings location. It is important to notice that other methods like the ray-tracing method need a much more complicated and computationally heavy preprocessing to obtain the layout in vector form. The result is a 2-dimensional

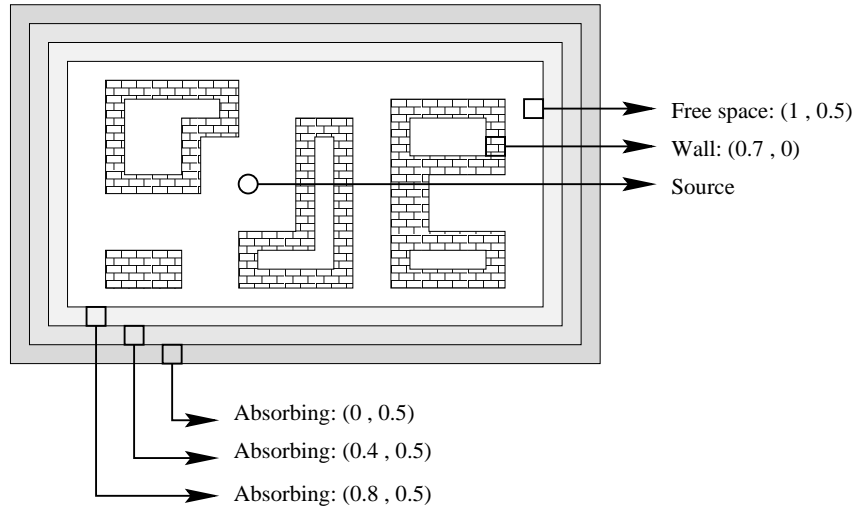


FIGURE 4.7: The different elements of the 2-dimensional discretized layout of a typical wave propagation experiment. The couple of coefficients (c_1, c_2) corresponding to (4.4) is given for each case.

array of typical size between 100×100 and 1000×1000 containing mainly four sorts of sites: the free space, the buildings and the absorbing frame, each one characterized by the couple of local parameters (c_1, c_2) and finally the source location defined by (γ, T) ; see figure 4.7 for a schematic presentation of the various elements. The evolution then consists of a synchronized updating followed by a nearest neighbors shifts of each site according to the evolution rules given in the preceding sections.

4.6.1 The Knife-wedge Test Problem

The simplest layout, after the vacuum case (sic), the so-called perfect absorbing knife-wedge problem, will be considered now to test our model. This is a theoretical problem because the physical reality of this type of absorber is questionable but it provides a useful and simple mathematical model for the study of diffraction by absorbing structures. The chosen layout is shown in the inset of figure 4.8: a plane wave arrives perpendicularly to a semi-infinite-line-shaped screen. Our screen is constituted of sites of reflection with $\mu = 0$, thus unperfect absorbers as defined in 4.6. The measurement is recorded along a line parallel to the source, behind the screen. We shall focus our attention on the shadow boundary, namely the region around $r = 0$.

The incident rays exists everywhere in the illuminated region $r > 0$. Upon hitting the wedge surface, an incident ray is completely absorbed, so no reflected rays exists. The ray striking the edge, however, is scattered in all directions

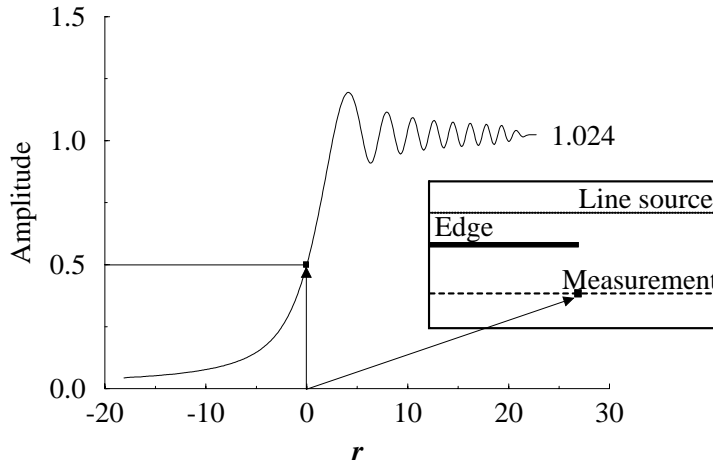


FIGURE 4.8: The “knife edge” test problem for a line source parallel to the half plane screen: the layout is shown in the inset of the figure. The measurement is performed along a line parallel to the line source, but behind the screen; r is given in wave length units. The amplitude at the shadow limit behind the edge should be half of the amplitude far from the edge. Here we have used $T = 16$ and the error is about 2%.

and gives rise to the spectrum of diffracted rays. The field amplitude along a diffracted rays excited by a unit amplitude incident ray is that one issued from a virtual unit amplitude point source located at the edge position multiplied by a geometric factor. However, it turns out that a simple ray-optical interpretation breaks down in the transition region $r \approx 0$ (the so-called shadow boundary): the above mentioned geometrical factor diverges and a much more complicated description involving a special transition factor must be employed. This is the key difficulty that must be handled when solving diffraction problems in inhomogeneous medium such as urban area.

As can be seen on figure 4.8 our method can solve the knife-wedge test problem with a good accuracy depending mostly on the choice of T , the source period. No special care must be taken and the expected diffraction pattern is directly obtained as a collective behavior of the lattice sites. In particular, the value of $1/2$ of the field amplitude on the transition point predicted by the wave theory is found with an error of 2% which could be reduced by the choice of a bigger period. Moreover this appears to be roughly independent of the reflection coefficient and other values for μ were equally good; it is purely the breaking of the spatial homogeneity that is relevant.

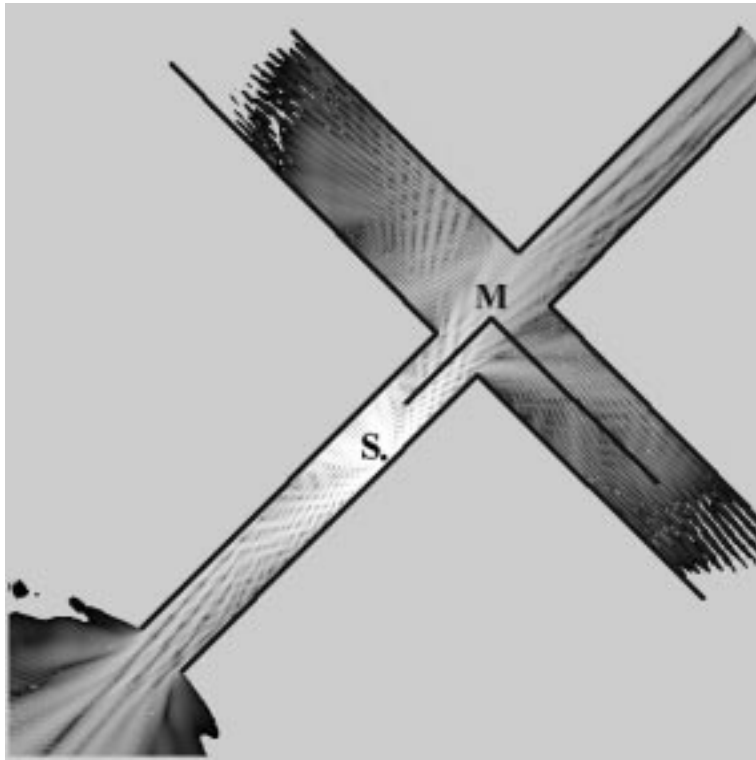


FIGURE 4.9: *Layout and result of simulation for the simple four-corner urban layout mapped on 600×600 sites. The source location is indicated by “S” and the measurement path by “M”. The curvilinear abscissa along the road starts near the source and is about 60 meters long. The various parameters used for this simulation are given in appendix C which illustrates how our simulation program Parflow is actually working. The gray levels are linearly distributed between 50 dB (white) and 100 dB (black). Comparison between prediction and real measurements are proposed in figure 4.10.*

4.6.2 The Four-corner calibration

The four-corner problem constitutes an important test for our model. The layout is simple and correspond to a real case for which real measurements performed by the Telecom PTT are available; see [46]. The overall configuration is shown on figure 4.9 but the details of the parameters is given in appendix C. The good agreement shown in figure 4.10 between measurement and simulation constitutes a first validation of our LB method for such problems. It is interesting to note that the renormalization is essential to catch the correct overall value as it corresponds to a shift of the resulting amplitude of about 40 dB.

The small scale intensity variations or fast deviations that are visible on figure 4.9 were averaged to obtain significant result as shown in 4.10. These fast deviations clearly come from the interference pattern resulting from our large

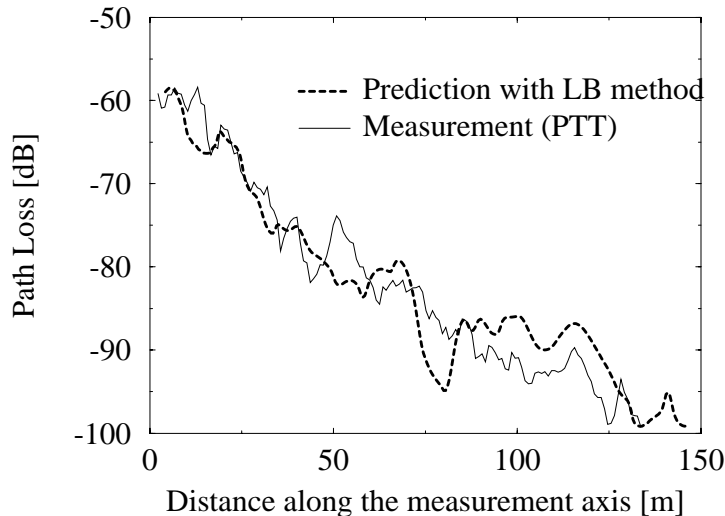


FIGURE 4.10: Quantitative result along the measurement axis corresponding to the layout shown on figure 4.9. The real in situ measurements performed by the TELECOM PTT (now SWISSCOM SA) are also shown for comparison. The good agreement between measurements and predictions in this case is a step forward in the test of our LB method. It also allows us to heuristically fix the reflection coefficient in further simulations.

simulation wave length. Fast deviations are also present in real measurement but at a much smaller scale so that averaging both prediction and measurement is necessary to provide significant comparisons.

Other simple configurations were investigated and intensive comparison between the LB method and the ray tracing method are presented in [47]. The conclusion is that it is sufficient to consider only the relevant 2-dimensional building layout especially near the transmitting antenna position, if the antenna is below the surrounding rooftops. However, the parameters affecting the reflection coefficient has to be adjusted to give correct prediction and ways to determine these parameters from the physical and measurable features of the buildings remain to be studied.

4.6.3 A Full Urban micro-cells

The situation described in figure 4.11 is fairly more complicated. A full micro-cell covering an urban area of size about 500×500 meters in the city of Bern, is investigated. As shown in figure 4.12, the features of our renormalized prediction, along the “Breitfeld street” fit well the real measurements. In particular, the overall value of the pathloss, the position and the shape of the peaks are relatively precisely captured by our simulations.



FIGURE 4.11: *Layout and result of simulation for the complete urban microcell located in the center of the city of Bern; the simulation arrays are of size 600×600 . “S” marks the source location and “M” the measurement axis. Gray scale is distributed between 30 dB (white) and 120 dB (black). The field intensity measurement is given in 4.12*

The remaining discrepancies between measurements and simulation are discussed below.

- A. The choice of the Manhattan distance in the renormalization process. As we saw in section 4.2, the Manhattan distance, while approximative for diagonal segments is well adapted to our problem because part of its anisotropy compensates the anisotropy of the propagation. But, roughly speaking, it appears that our predictions are less contrasted than the real field: measured peaks and minima are more pronounced; we believe that a better distance estimation, for instance based on the mean delay time of the impulse response will contribute to a better discrimination between point nearby and far-off the source.
- B. The choice of the orientation of the layout with respect to lattice directions. The effect of the lattice orientation is completely arbitrary and may introduce errors up to 10 dB. A computationally heavy way of overcoming this

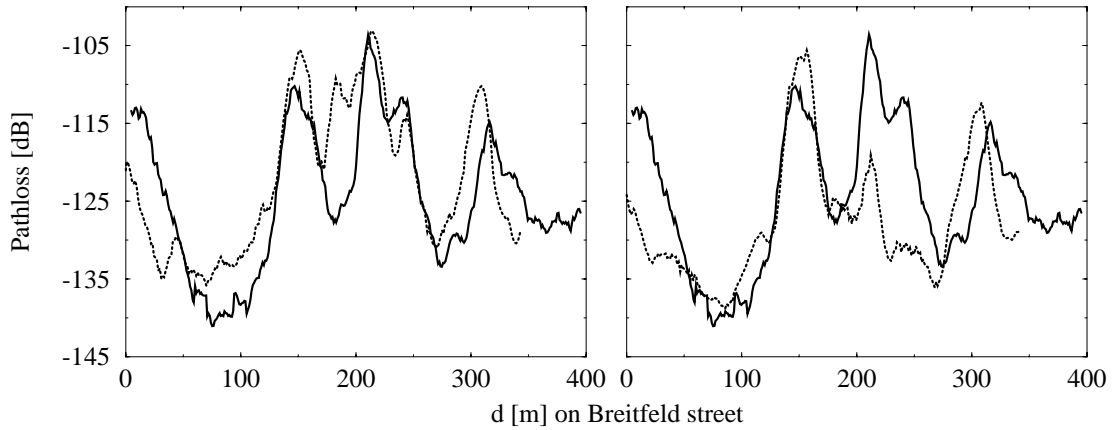


FIGURE 4.12: Quantitative result along the measurement axis corresponding to the layout shown in figure 4.11 (city of Bern). Two types of boundary condition were applied for the sites representing the buildings in the discretized layout: sites of reflection with $\mu = 0.39$ (right part of the figure) resulting in opaque buildings and sites of attenuation with $\mu = 0.39$ (left part of the figure) corresponding to permeable buildings in order to account for the propagation above the roofs.

error source could be to average over different orientation because none of them should *a priori* be privileged.

- C. The choice of the wall microscopic properties. It has been already mentioned that the parameter c_1 affecting the reflection coefficient has to be adjusted to give correct prediction but a different choice for c_2 may also be possible. Indeed a $c_2 \neq 0$, e.g. permeable buildings may be a valuable technique to take into account two limitations of our model, namely the penetration of the waves in the narrow streets when their width become comparable to the simulated wave length and the 3-dimensional contribution of the waves arriving at observation point from above the rooftops.
-

Other Applications

The applications presented in this chapter are based on the same microscopic model as used for simulating radio wave propagation. At first sight, the situations are however completely different and it is noteworthy that the same approach, *i.e.* almost the same numerical algorithms, may address such a wide variety of different physical phenomena.

Section 5.1 present a CA-model for large-solids. Our CA-solid may undergo a global motion mediated by propagation of small deformations. We can make use of our microscopic wave model because these deformations spread like waves throughout the solid. Where the deformations exceed a threshold value we shall locally break the bounds. This technics, which turns out to be a valuable approach for simulating crack propagation, is presented in sub-section 5.1.4.

As for section 5.2, it addresses the problem of wave localization. The idea is that a wave may lose all its propagating properties when evolving in a random media. The problem is subtle since the phenomena associated with wave localization are of a kind between wave propagation and diffusion. Also in this case, our approach provides us with a promising numerical and theoretical tool.

5.1 Large-scale Moving and Cracking Objects

Modeling large scale objects with CA and lattice Boltzmann approach has remained mostly unexplored. A large scale-scale object is a virtual coherent ensemble of many particles, or “atoms” whose size is greater than the range of the CA rules, typically greater than the inter-particles spacing. The coherence requirement concerns (i) the capability for these objects to undergo a global motion in space, while preserving their overall shape, (ii) to assume the propagation

of local deformations without the components spreading out in the entire space, and (iii) to be characterized by macroscopic, appropriately defined, quantities like mass, energy and momentum possibly conserved under certain conditions.

Models for large scale objects provide interesting possibilities for simulating new complicated physical situations. One of the most challenging physical problem that may be addressed, is the study of fracture process; see the attempt [48]. Recently, large-scale molecular dynamics[49] simulations have shown that atomistic models were able to predict new phenomena like fracture instabilities or the abnormal limiting crack velocity, not embeded in the classical continuum picture that has historically provided, and is still providing in numerous engineers applications, most of the theoretical tools.

However the need for a simple model is still of great importance for studying open basic questions. Whereas molecular dynamics simulations is endlessly trying to approach the improbable perfect adequation between reality and model in terms of interactions, nature and number of atoms involved¹, thus costing each day even more memory and calculation time that are increasingly provided by actual super computers, our LB approach remains simple and straightforward and is promising because of its global behavior, although not strictly realistic at the atomic level.

This section is firstly recalling the successful attempt to model 1-dimensional objects called strings as a CA, described in [51]. Next, the 2-dimensional model is proposed and turns out to be a reinterpretation of the wave automaton presented in our previous chapters. Finally, two applications are presented: the solid body motion with external forces like reflecting boundaries or gravity field and the dynamics of the fracture of a pre-loaded sheet.

5.1.1 The 1-Dimensional CA String Model

The string presented in [51] can be thought of as a chain of masses linked by springs. More precisely it is composed of two kinds of particles, say the white ones and the black ones, which alternate along a line. In the initial configuration the particles are positioned on a regular 1-dimensional lattice and two successive particles are either nearest neighbors or separated by exactly one lattice spacing. The CA rule may easily be generalized to handle more than one exact lattice spacing (2, 3, 4... for instance) between particles.

The rule of motion is shown in figure 5.1: black and white particles are moving alternatively each other time step and the new configuration is obtained by interchanging the lattice spacings that separate the moving particles from their

¹Current 3-dimensional Molecular dynamics simulations, like those performed at Cornell University on IBM RS/6000 Scalable POWERParallel Systems (SP), may involve up to 100 millions atoms interacting through a two-body Lennard-Jones potential. The simulation of 1 nanosecond crack propagation takes up to 100 hours of calculation; see the report [50]

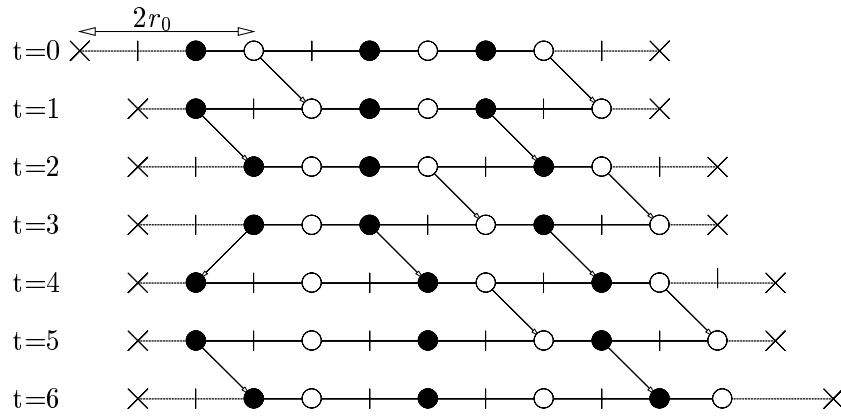


FIGURE 5.1: Typical motion of a 1-dimensional string on a lattice. White particles are moving first. The cross (\times) indicates the position of the virtual particles necessary to calculate the motion of the 2 boundary particles. It is positioned at a fixed distance $2r_0$ away from the nearest neighbor of the corresponding boundary particle; here r_0 equal $3/2$ lattice spacing. This configuration will need exactly 10 time steps to cycle back to its initial configuration. Meanwhile it will travel 3 lattice spacings; therefore its overall speed is 0.3.

adjacent neighbors at rest. In other words the move of a particle corresponds to a reflection with respect to the center of mass of its two neighbors.

Of course, this rule is not valid for the end particles since they only have one neighbor. In fact, their motion is governed by the very same dynamics provided that a virtual or pseudo-particle is added at the extremity of the chain. The virtual particles is placed at a fixed distance $2r_0$ away from the next to last string particle. The constant r_0 corresponds to the length at rest of the spring connecting the particles. With this particular rule for the boundary we can see that the equilibrium configuration (all particles at rest) is achieved when all inter-particles distance equal r_0 .

We shall demonstrate now that the basic feature of this rule is to propagate perturbations along the chain. We define $f_1(i)$, $f_2(i)$ as the oriented deformation of the springs for those particles which are moving, say the white, and the constant r_0 for the others. More precisely we have (beware of the sign):

$$f_1(i) = \begin{cases} x(i+1) \Leftrightarrow x(i) \Leftrightarrow r_0 & \text{if particle } i \text{ is moving now} \\ 0 & \text{if particle } i \text{ do not move} \end{cases}$$

$$f_2(i) = \begin{cases} x(i \Leftrightarrow 1) \Leftrightarrow x(i) + r_0 & \text{if particle } i \text{ is moving now} \\ 0 & \text{if particle } i \text{ o not move} \end{cases}$$

where $x(i)$ is the coordinates of the i 's particle of a N -atoms string: $i = 1, \dots, N$.

We write $f_\alpha^*(i)$ any $f(i)$ involving the coordinate of a pseudo-particles. We have

$$\begin{aligned} f_1^*(N) &= x(N \Leftrightarrow 1) + 2r_0 \Leftrightarrow x(N) \Leftrightarrow r_0 \\ &= f_2(N) \end{aligned}$$

and similarly

$$\begin{aligned} f_2^*(1) &= x(2) \Leftrightarrow 2r_0 \Leftrightarrow x(1) + r_0 \\ &= f_1(1) \end{aligned}$$

Thus the evolution rule may be written like following:

$$\begin{pmatrix} f_1(i \Leftrightarrow 1, t+1) \\ f_2(i+1, t+1) \end{pmatrix} = \begin{cases} \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} f_1(i, t) \\ f_2(i, t) \end{pmatrix} & \text{if } i \text{ is inside the string,} \\ \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} f_1(i, t) \\ f_2(i, t) \end{pmatrix} & \text{else.} \end{cases} \quad (5.1)$$

With this formulation, it is clear that the string deformation is mediated by the propagation of forward and backward waves, f_1 and f_2 respectively. The boundary conditions are reflection of these waves. The initial condition, together with the definition of r_0 completely determine the dynamics of the string.

To express this rule as a true CA, the particles must remain on the lattice (integer positions) while moving. We show that this is achieved by a particular choice of r_0 . We name k_i 's the initial positive integer distances (or sites intervals) between the particles of a chain. During a time step of evolution, k_i transforms in k'_i , but k'_i remains integer for all the particles inside the string. Indeed, the k_i 's are simply exchanged by the dynamics. As for the boundaries - or end-particles - we have:

$$k \Leftrightarrow k' = 2r \Leftrightarrow k, \text{ where } k \in \mathbb{N}^*$$

where we must take care that k' remains integer and strictly positive to ensure the CA-like dynamics and the conservation of the string integrity. If k_0 the maximum inter-particle distance present in the initial configuration is $k_0 = 2$, we must choose $r_0 \geq 3/2$ with $2r_0 \in \mathbb{N}^*$. In figure 5.1 we have chosen $k_0 = 2$ and $r_0 = 3/2$. It is interesting to note that no "frozen" chain may exist on a lattice when $k = 0$.

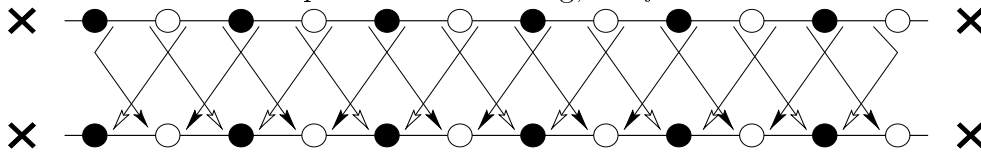
The deformation energy, in other words the potential energy due to the deformation of the springs in a N -atoms string is given by

$$\begin{aligned} E &= \frac{\kappa}{2} \sum_{i=1}^{N-1} (x(i+1) \Leftrightarrow x(i) \Leftrightarrow a)^2 \\ &= \frac{\kappa}{2} (f_1^2(1) + f_2^2(N)) + \frac{\kappa}{2} \sum_{i=2}^{N-1} f_1^2(i) + f_2^2(i) \end{aligned} \quad (5.2)$$

where κ is an arbitrary spring constant. Recall that $f = 0$ for rest particle. In other word, E is the sum of all the “real” $f_\alpha^2(i)$ *e.g.* to the exclusion of the virtual f_α^* . The total momentum of a string is the sum of the mass multiplying the speed of the particle, i.e. the distance it will travel during the next time step (note that the end particles have a mass $1/2$, as a convenient choice for reasons explained in [51]):

$$\begin{aligned}
 P &= \sum_{i=1}^N m(i)(f_1(i) + f_2(i)) \\
 &= \frac{1}{2}(f_1(1) + f_2^*(1)) + \sum_{i=2}^{N-1} f_1(i) + f_2(i) + \frac{1}{2}(f_1^*(N) + f_2(N)) \\
 &= f_1(1) + f_2(N) + \sum_{i=2}^{N-1} f_1(i) + f_2(i)
 \end{aligned} \tag{5.3}$$

Thus ρ is the sum of all the “real” $f_\alpha(i)$. Of course, we have mass conservation during the evolution. To justify our definition of energy and momentum we will show that these quantities are conserved during the evolution of the string, as expected for a solid body not submitted to external forces. With the definition of the virtual particles, the evolution rules is just a reordering of all the f ’s as shown in the scheme below. A time step evolution of the string is schematically represented, and one must imagine each particles having a f_1 on its right side and a f_2 on its left side. Here, the f_1 of a white particle has the same value as the f_2 of the next to the right black neighbors. Suppose the white particles will move the next time step, as a result, the quantities f ’s will move following the white arrows. If the black particles are moving, the f will follow the black arrows:



So we have the conservation of each individual f except $f_2^*(1)$ and $f_1^*(N)$ which are not present in (5.2) and (5.3) and consequently we conserve E , P . Note that these conservation laws apply for an even more general situation where all particles may move at the same time. This case was not mentioned here in the context of the string motion because it may break the integrity of our chain by interchanging the order of the particles.

If f are real-valued quantities, we observe that the evolution rule given in (5.1) is equivalent to 1-dimensional lattice Boltzmann wave model to be found in appendix A. In the next section we will show that our 2-dimensional wave automaton is also equivalent to a generalization of the string model: the sheet model.

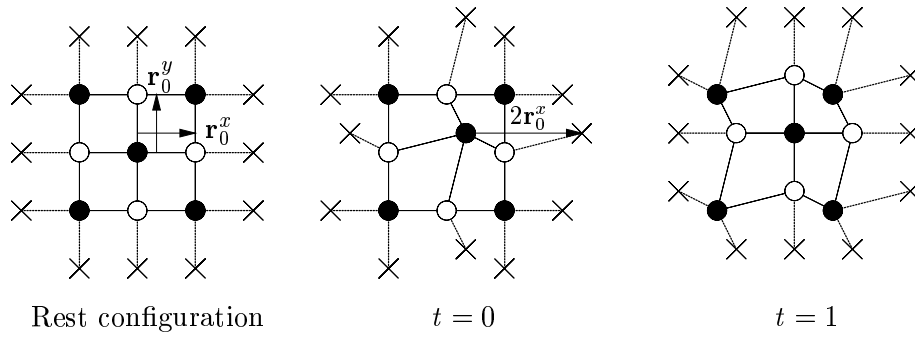


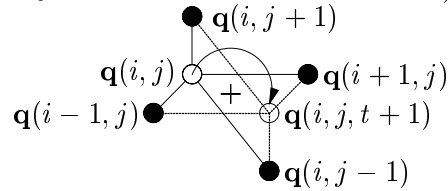
FIGURE 5.2: Example of a sheet of black and white atoms of size 3×3 . The cross (\times) indicates the virtual particles added to account for the boundaries of the sheet. The rest configuration is shown on the left and a time step of evolution (whites are moving between $t = 0$ and $t = 1$) can be seen on the right.

5.1.2 The 2-Dimensional Sheet model

Our 2-dimensional sheet of “atoms” is a square lattice of particles, each of them connected with its four neighbors. At rest, the position of the atoms coincide with a finite regular square lattice; with lattice spacing r_0 , as shown in figure 5.2. Black and white particles alternate along each dimension. During the evolution, deformations may propagate in the sheet, but the topology remains the same, e.g. the identity of each atoms and its neighbors does not change. Consequently, we call $\mathbf{q}(i, j)$ the spatial coordinates of the atom belonging the i 'th row and j 'th column of the sheet. In the same spirit as for the spring model, we define the deformation vectors \mathbf{f}_α :

$$\begin{aligned} \mathbf{f}_1(i, j) &= \mathbf{q}(i + 1, j) \Leftrightarrow \mathbf{q}(i, j) \Leftrightarrow \mathbf{r}_0^x \\ \mathbf{f}_2(i, j) &= \mathbf{q}(i, j + 1) \Leftrightarrow \mathbf{q}(i, j) \Leftrightarrow \mathbf{r}_0^y \\ \mathbf{f}_3(i, j) &= \mathbf{q}(i \Leftrightarrow 1, j) \Leftrightarrow \mathbf{q}(i, j) + \mathbf{r}_0^x \\ \mathbf{f}_4(i, j) &= \mathbf{q}(i, j \Leftrightarrow 1) \Leftrightarrow \mathbf{q}(i, j) + \mathbf{r}_0^y \end{aligned}$$

where $\mathbf{r}_0^x = (r_0, 0)$ and $\mathbf{r}_0^y = (0, r_0)$. The evolution has the same flavor as for the string model; the moving particles (black ones and white ones, alternatively) jump to the symmetrical position with respect to the center of mass g of its neighbors (indicated by a cross in the scheme below):



More precisely we have the following dynamics:

$$\begin{aligned}
\mathbf{q}(i, j, t + 1) &= \mathbf{q}(i, j, t) + 2(\mathbf{q}_g(i, j) \Leftrightarrow \mathbf{q}(i, j, t)) \\
&= \mathbf{q}(i, j, t) + 2\left(\frac{1}{4}(\mathbf{q}(i + 1, j) + \mathbf{q}(i, j + 1) \right. \\
&\quad \left. + \mathbf{q}(i \Leftrightarrow 1, j) + \mathbf{q}(i, j \Leftrightarrow 1)) \Leftrightarrow \mathbf{q}(i, j, t)\right) \\
&= \mathbf{q}(i, j, t) + \frac{1}{2} \sum_{\alpha=1}^4 \mathbf{f}_\alpha(i, j)
\end{aligned}$$

Consequently, the evolution of, say, \mathbf{f}_1 is given by

$$\begin{aligned}
\mathbf{f}_1(i \Leftrightarrow 1, j, t + 1) &= \mathbf{q}(i, j, t + 1) \Leftrightarrow \mathbf{q}(i \Leftrightarrow 1, j, t + 1) \Leftrightarrow \mathbf{r}_0^x \\
&= \mathbf{q}(i, j, t) \Leftrightarrow \mathbf{q}(i \Leftrightarrow 1, j, t + 1) \Leftrightarrow \mathbf{r}_0^x + \frac{1}{2} \sum_{\alpha=1}^4 \mathbf{f}_\alpha(i, j) \\
&= \mathbf{q}(i, j, t) \Leftrightarrow \mathbf{q}(i \Leftrightarrow 1, j, t) \Leftrightarrow \mathbf{r}_0^x + \frac{1}{2} \sum_{\alpha=1}^4 \mathbf{f}_\alpha(i, j) \\
&= \frac{1}{2}(\mathbf{f}_1(i, j) + \mathbf{f}_2(i, j) \Leftrightarrow \mathbf{f}_3(i, j) + \mathbf{f}_4(i, j))
\end{aligned}$$

Similarly, the other \mathbf{f} may be determined and we find the usual TLM matrix \mathbf{W} define in 2.52, excepted that \mathbf{f} has now 2 independent dimensions. Note also that the \mathbf{f} 's are moving from one site to another with the reverse speed compared to the definitions given in the previous chapter.

$$\begin{aligned}
\begin{pmatrix} \mathbf{f}_1(i \Leftrightarrow 1, j, t + 1) \\ \mathbf{f}_2(i, j \Leftrightarrow 1, t + 1) \\ \mathbf{f}_3(i + 1, j, t + 1) \\ \mathbf{f}_4(i, j + 1, t + 1) \end{pmatrix} &= W_{\alpha\beta} \mathbf{f}_\beta \tag{5.4} \\
&= \frac{1}{2} \begin{pmatrix} 1 & 1 & \Leftrightarrow 1 & 1 \\ 1 & 1 & 1 & \Leftrightarrow 1 \\ \Leftrightarrow 1 & 1 & 1 & 1 \\ 1 & \Leftrightarrow 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} \mathbf{f}_1(i, j, t) \\ \mathbf{f}_2(i, j, t) \\ \mathbf{f}_3(i, j, t) \\ \mathbf{f}_4(i, j, t) \end{pmatrix}
\end{aligned}$$

The evolution given here applies only for atoms inside the sheet. For boundary atoms, some \mathbf{f} 's are not defined and the evolution rule must specialized. Here again, we will add “virtual” particles where links are absent; see the crosses (\times) on figure 5.2. The virtual particle is added in such a way that the missing link, say \mathbf{f}_1^* is equal to the opposite \mathbf{f}_3 . In the \mathbf{q} variables it means

$$\begin{aligned}
\mathbf{f}_1^* &= \mathbf{f}_3 \\
\mathbf{q}^*(i + 1) \Leftrightarrow \mathbf{q}(i) \Leftrightarrow \mathbf{r}_0^x &= \mathbf{q}(i \Leftrightarrow 1) \Leftrightarrow \mathbf{q}(i) + \mathbf{r}_0^x \\
\mathbf{q}^*(i + 1) &= \mathbf{q}(i \Leftrightarrow 1) + 2\mathbf{r}_0^x
\end{aligned}$$

where the * indicates the virtual attribute of the associated variable. This definition is enough to simulate the motion of a square-like sheet in a rigid container as will be presented in the next section.

Of course the dynamics of our system conserves the number of particles and thus the total mass $M = \sum_{ij} m(i, j)$, where $m(i, j)$ is the mass of each individual atoms. The m_i 's indicate not a physical mass in the strict sense of the word. It is rather a coefficient that will turn out to be necessary to distinguish the inside- and boundary-particles. Consequently, the m_i 's are more likely *weights* in the statistical sense than *masses* in the physical sense.

The need to distinguish the inside- and boundary-particles arise when we want to conserve more interesting macroscopic quantities such as energy and momentum. More precisely, we must set different weights for inside-, side- and corner particles. It is possible to show that the adequate choice is as follows:

$$m(i, j) = \begin{cases} 1 & \text{For inside particles} \\ \frac{1}{2} & \text{For boundary particles} \\ \frac{1}{4} & \text{For the corners} \end{cases} \quad (5.5)$$

Thus, the conserved total energy of the sheet, in the absence of external forces, is given by the following weighted sum of all the square deformation:

$$E = \frac{1}{2} \sum_{i,j} m(i, j) \sum_{\alpha} \left(\frac{\mathbf{f}_{\alpha}(i, j)}{2} \right)^2$$

where all \mathbf{f} 's (virtual or real) are taken into account. It is interesting to note that another definition of E is also possible: instead of using the mass $m(i, j)$ we could use a different factor for each \mathbf{f}_{α} . To obtain an equivalent expression for E we must count each boundary \mathbf{f}_{α} for half of its value and each virtual \mathbf{f}_{α}^* for nothing.

The kinetic energy E_{kin} is defined by the motion of the center of gravity G :

$$\begin{aligned} E_{\text{kin}} &= \frac{1}{2} M v_G^2 \\ &= \frac{1}{2M} \left(\sum_{i,j} m(i, j) (\mathbf{q}(i, j, t+1) \Leftrightarrow \mathbf{q}(i, j, t)) \right)^2 \\ &= \frac{1}{2M} \left(\sum_{i',j'} \frac{1}{2} m(i', j') \sum_{\alpha} \mathbf{f}_{\alpha}(i', j') \right)^2 \\ &\doteq \frac{1}{2M} \left(\sum_{i',j'} \mathbf{p}(i', j') \right)^2 \doteq \frac{\mathbf{P}^2}{2M} \end{aligned} \quad (5.6)$$

where the prime indicates that the indices are only those of the particles which are moving during the next step; thus, the corresponding sums run over half of

the atoms. Equation 5.6 defines the total momentum \mathbf{P} of the sheet and $\mathbf{p}(i', j')$, the momentum of a moving particles.

The total energy in absence of external forces can be seen as composed of one purely internal part which may be at the origin of the “temperature” of the sample and one kinetic energy responsible for the global motion of the sheet. So we may give the following definition for the internal energy:

$$E_{\text{int}} \doteq E \Leftrightarrow E_{\text{kin}}$$

Consequently we can define the temperature of our solid as being $T \doteq E_{\text{int}}/M$. Our definition of the temperature is consistent with the equipartition theorem² providing that M is interpreted as the total number of degrees of freedom of our solid. The equipartition theorem allows us to re-interpret the “masses” or “weights” m_i defined in (5.5): m_i is proportional to some the degree of freedom of a particle; this is coherent with fact that we had to impose different m_i ’s for side- and corner-particles.

Note that the interaction we have considered between black and white particles do not correspond exactly to that of a regular spring: we have here completely decoupled the x and y motions which is not the case with a spring where the force depends on the Euclidean distance between the pairs of particles. However, we shall see that essential features of a simple solid are captured by our model with much less complication.

5.1.3 Solid Body Motion

The purpose of this section is to simulate the motion of a 2-dimensional deformable solid of connected atoms in a rigid container. The model developed in the previous section will be stated more specifically in order to take into account the impassable boundary of the limiting space and a possible external volume force like the gravity. The initial configuration fully determines the subsequent evolution of the sheet which momentarily deforms but preserves its integrity during the collision processes as suggested on figure 5.3.

Starting from the rest position \mathbf{q}_0 of the atoms an initial configuration may be prepared with a specific macroscopic momentum \mathbf{P} and a temperature T . If we add \mathbf{v}_0 to the position of every particle which will move the next time steps,

²The equipartition theorem of the statistical physics theory states that a particle in a bulk has an average energy proportional to the temperature, and proportional to its number of degrees of freedom: $kT/2$ per degree of freedom, where k is the Boltzmann constant.

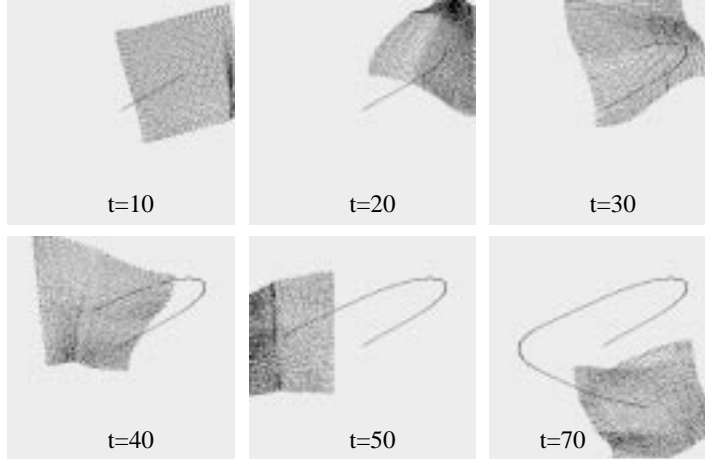


FIGURE 5.3: Motion of a deformable object made of a square structure of particles interacting according to the lattice Boltzmann wave rule. The solid bounces back on the boundaries of the simulation.

we obtain:

$$\begin{aligned}
 \mathbf{P} &= \sum_{i',j'} \mathbf{p}(i',j') = \sum_{i',j'} \frac{1}{2} m(i',j') \left(\sum_{\alpha} \mathbf{f}_{\alpha}(i',j') \Leftrightarrow \mathbf{v}_0 \right) \\
 &= \sum_{i',j'} \Leftrightarrow 2m(i',j') \mathbf{v}_0 \\
 &= \Leftrightarrow M \mathbf{v}_0
 \end{aligned}$$

where we have supposed, for simplicity, that $\sum_{i',j'} m(i',j') = M/2$, e.g. the number of black and white particles are equal. In other words, the macroscopic momentum \mathbf{P} is equal to the microscopic initial (homogeneous) deformation times the total mass. Note that if we have added \mathbf{v}_0 to the other sub-lattice (the particles at rest) this would only change the sign of \mathbf{P} . This kind of deformation gives rise to a pure translation, thus we expect to find $E = E_{\text{kin}}$:

$$\begin{aligned}
 E_{\text{kin}} &= \frac{\mathbf{P}^2}{2M} = \frac{1}{2} M \mathbf{v}_0^2 \\
 E &= \frac{1}{2} \sum_{i,j} m(i,j) \frac{1}{4} \sum_{\alpha} \mathbf{v}_0^2 \\
 &= \frac{1}{2} M \mathbf{v}_0^2 = E_{\text{kin}} \quad \square
 \end{aligned}$$

Now, suppose we add a random variable, e.g. a kind of noise, $\mathbf{n}(i,j)$ to every atom's position. The variable \mathbf{n} follows a normal centered distribution $\mathcal{N}(0, \sigma)$, where σ is the standard deviation. The momentum $\mathbf{P} = \Leftrightarrow M \mathbf{v}_0$ is not changed

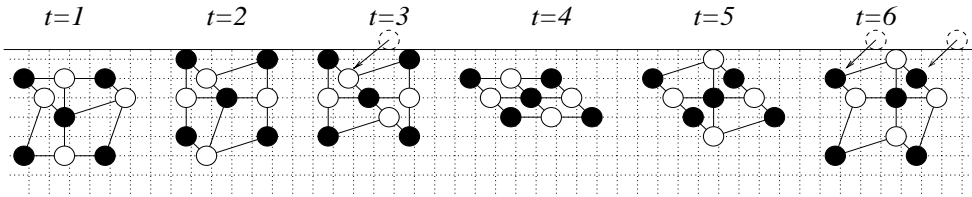


FIGURE 5.4: Reflection of a 3×3 sheet against a wall (r_0 is 2 lattice spacing). The black particles are moving at odd time steps and the arrows shows the particles kept in their original position because of an impossible move. We can observe that the configuration at $t = 6$ is similar (equal providing a space inversion plus a $\pi/4$ rotation) to those before the collision. It is interesting to note that, in this special case, the solid never jumps off lattice during the process.

and the energy E is given by:

$$E = \frac{1}{2} \sum_{i,j} m(i,j) \frac{1}{4} \sum_{\alpha} (\mathbf{v}_0^2 + \mathbf{n}^2(i,j) + 2\mathbf{v}_0 \mathbf{n}(i,j))$$

$$\approx E_{\text{kin}} + \frac{1}{2} M 2\sigma^2 + 0 \implies T = \sigma^2$$

The approximation is only due to the mass difference between inside and boundary particles (see (5.5)) and the equality is restored for infinitely big sheet. In other words the temperature of a LB solid is equal to the variance of the normal noise on the particle position. Due to the propagation of these initial deformations, our solid is now moving in space with a specific momentum \mathbf{P} and temperature T .

Our solid is now able to move in a continuous space; the position are no longer restricted to integer numbers. Next we shall define what happened on the boundary of the simulation space which can be now imagined as box containing our solid. The reflection of the solid projected against a rigid wall will be treated at the level of the individual particles. The rule to apply is again simple: if a particle wants to jump in a prohibited area, e.g. on the other side of the wall, the particle is kept on its original position before the jump; as shown in figure 5.4. Thus, the evolution matrix W defined in equation 5.4 becomes: $W = \mathbf{1}$ and consequently we are sure to conserve the total energy E . On the contrary E_{kin} is not necessary conserved as will be shown later. The time necessary to complete a reflection corresponds to the time taken by the deformation to travel twice through the sample (back and forth). Remembering that the propagation in a TLM dynamics occurs at speed $c = 1/\sqrt{2}$ we have

$$t_{\text{ref}} = 2N\sqrt{2}$$

Figure 5.3 and 5.5 show a typical motion inside a rigid box. During the collisions with the boundaries of the simulation the sheet bounce back but keeps

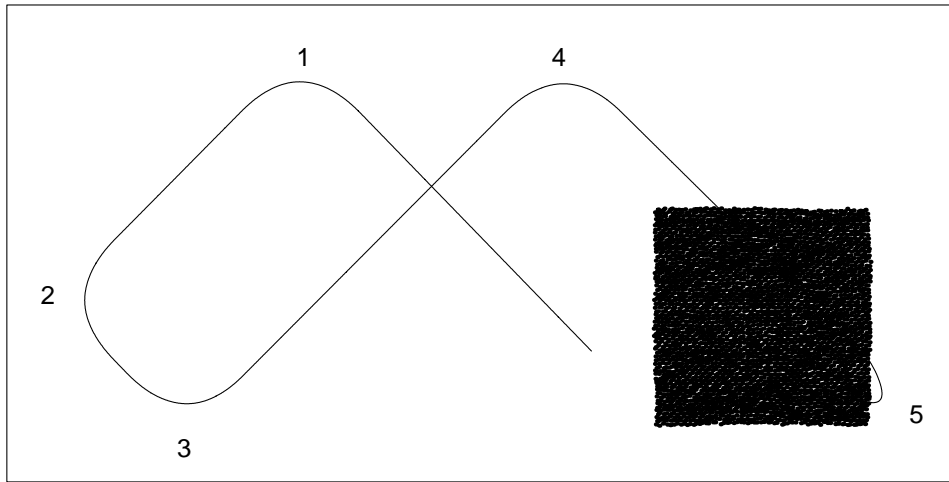


FIGURE 5.5: *Evolution of a 2-dimensional deformable sheet of “atoms” of size 64×64 without external volume force ($M=3969$). The initial configuration was prepared with a small noise normally distributed around the rest position: $\mathbf{q}(i, j) = \mathbf{q}(i, j) + \mathcal{N}(0, 0.1)$ and a global momentum: $\mathbf{q}_{black} = \mathbf{q}_{black} + (0.2, 0.2)$ (\mathbf{q} is given here without dimension, expressed in units of r_0). The box is perfectly reflecting the sheet as can be seen by following the trace of the center of mass (solid line). During this reflection processes, numbered from 1 to 5, the initial orientation of the solid is not changed. A part of the kinetic energy is converted in heat without changing the total amount of energy (see figure 5.6).*

its integrity. The energy is conserved throughout the evolution as can be seen on figure 5.6. The same experiment may be repeated with different initial temperatures. However we can observe that the internal energy, thus precisely the temperature, is not conserved during the reflection process, but remain constant between two collisions (as expected by the conservation of the momentum in absence of any external force). During the collision process, the solid is compressed, e.g. heated, until the information has been transferred to the entire system and then expanded, e.g. cooled again to approximately its original temperature. However, we can observe that a small part of the kinetic energy is dissipated in heat: the temperature increase slightly after each collision, see figure 5.6. This apparent “dissipation” comes from the fact each of the uncoupled x - and y -deformations actually does propagate in any direction. Thus, part of the deformations resulting from a collision process will propagate transversally and will thus never reach the opposite side of the solid. As a consequence, it will add up to the ambient noise instead of contributing to the overall motion.

It is interesting to note that this apparent macroscopic “dissipation” does not mean that our model is not time reversible. Actually, our model is microscopically time reversible. The time inversion is simply achieved by breaking once the

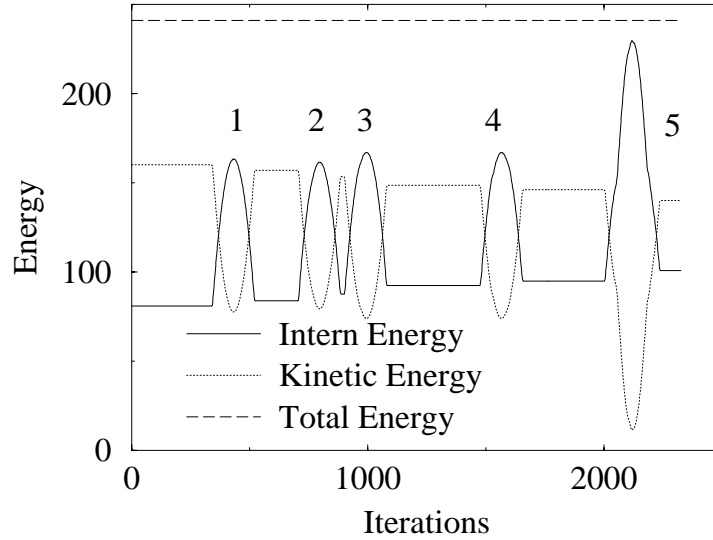


FIGURE 5.6: Evolution of the energies E_{int} and E_{kin} for the case presented in figure 5.5 where the number 1–5 correspond to the reflections indicated in 5.5. The temperature $T \neq 0$ at time $t = 0$ is due to the presence of noise in the initial configuration. The plateau correspond to the evolution in straight lines between two collisions. We can observe an important heat up-and-down of the sheet during the collision processes. Macroscopically, the collision is not reversible and the temperature (the disorder) is slightly growing after each reflection.

black & white alternation *e.g.* by moving the same subset of atoms during two consecutive time steps. If this is done, the solid returns exactly along his path and eventually finds again its original configuration in terms of the position of each atoms. Of course, during this way back, each collision will decrease the temperature proving that there is nothing intrinsically dissipative in the model.

Now, we will present how to introduce a gravity field for our system. This last refinement is given here to show the ease with which new features may be introduced in our model. Following our philosophy, the effect of an external volume force is considered at the microscopic level giving rise to the expected behavior at the macroscopic scale. The dynamics of the particles given in equation 5.4 is transformed as follows:

$$\mathbf{q}(i', j', t + 1) = \mathbf{q}(i', j', t) + \frac{1}{2} \sum_{\alpha=1}^4 \mathbf{f}_{\alpha}(i', j') + \mathbf{g}$$

This means not that a constant velocity is imposed to the particles. Indeed, the $\mathbf{f}_{\alpha}(i', j')$'s also contain the \mathbf{g} -contribution of the preceding time step. Now, we shall see that this new dynamics does actually correspond to an accelerated

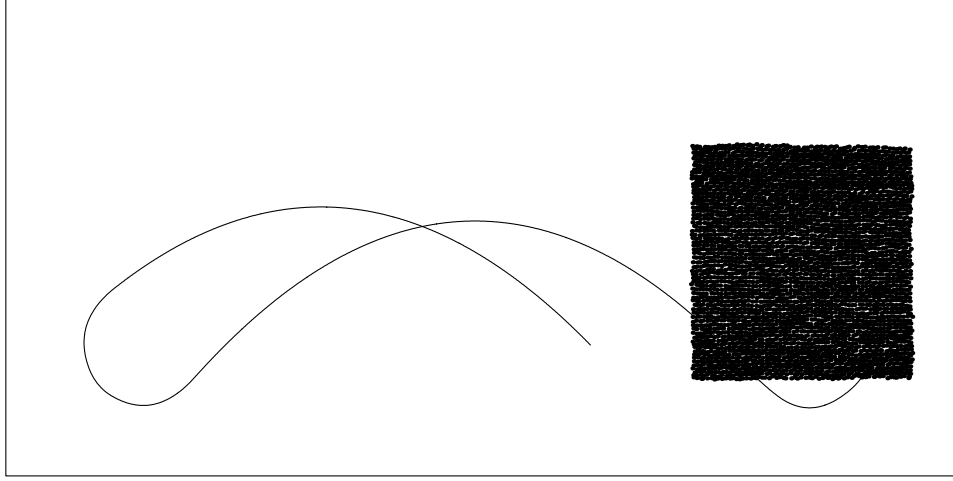


FIGURE 5.7: Trajectory of a 64×64 atoms sheet in a gravitational potential $\mathbf{g} = (0, 0.001)$. The initial condition are otherwise the same as in figure 5.5. The total energy is conserved: $E_{kin} + E_{int} + H$ and smooth parabolic trajectory take place between the rebounds.

motion in a constant force field. The total momentum is given by

$$\mathbf{p}(t) = \sum_{i'j'} m(i', j') \frac{1}{2} \sum_{\alpha} \mathbf{f}_{\alpha}(i', j', t) + \frac{M\mathbf{g}}{2}$$

This means that the momentum is increasing at each time steps by the amount of $M\mathbf{g}/2$. Recording the second Newton's law we infer that $M\mathbf{g}/2$ is a force and the potential energy introduced by the mean of \mathbf{g} is

$$H(\mathbf{Q}) = \frac{1}{2} \mathbf{Q}\mathbf{g}$$

where \mathbf{Q} is the position of the center of gravity of our solid. The figure 5.7 shows the behavior of our solid in a gravitational potential. We can appreciate the coherence of the definitions: the total energy $E_{int} + E_{kin} + H$ is conserved during the evolution. Observe in figure 5.8 that the gravity itself does not increase the temperature and the trajectory is a smooth parabola between two rebounds.

5.1.4 Fracture process

Fractures is a huge engineering problem: for instance, the cost of material failure in the United States has been estimated to about 4% of the national product in 1983. This has been a long standing problem, undoubtedly still increasing, and even Gallileo Gallilei in his *Dialogues Concerning Two New Sciences* (1646) presents the study of the forces that hold objects together and the conditions

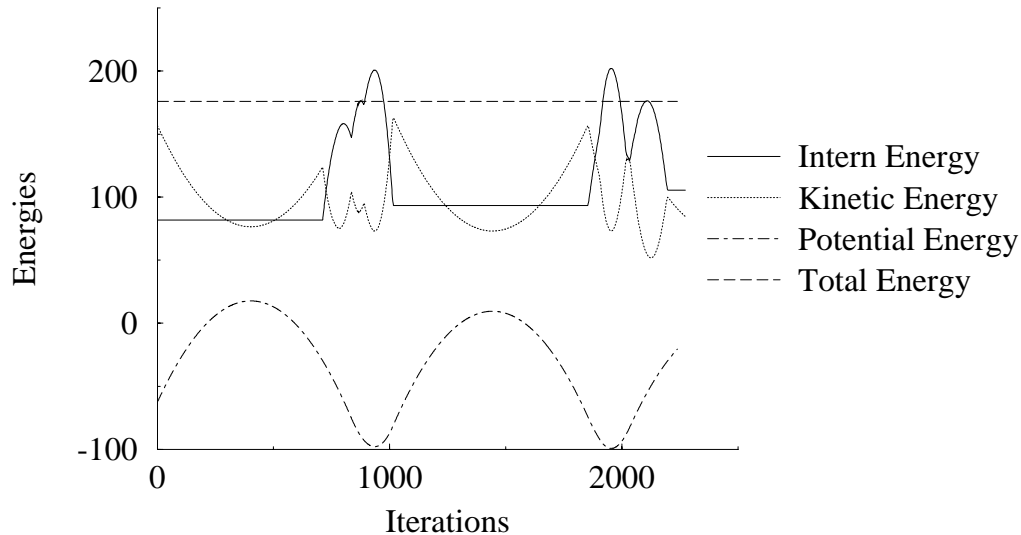


FIGURE 5.8: *Plot of the energies of the moving sheet in a gravity field shown in figure 5.7. Kinetic + Intern + Potential energy are conserved. Here again the temperature is slightly growing at each collision.*

that cause them to fall apart as his first new science. The second concerned the laws of motion governing the movement of projectiles, a science which has made significant progress ! Today, despite the tremendous development of solid-state physics, physicists have paid relatively slight attention to how things breaks[52].

For lack of a better understanding, engineers often view fracture mechanics as a macroscopic phenomenon and perform numerous bending and stretching tests using large machines to learn about material strength. However, the classical continuum picture is unsatisfactory and today's technological or fundamental questions require an atomistic understanding. But unfortunately, the relation between bounding energies and strength of materials is far from direct and, for instance, there is no completely satisfactory answer to the question of why some materials are brittle and other ductile.

A brittle material under a critical stress, with a small saw cut, will crack along a sharp line as if you were using a knife blade one atom wide. On the other hand, in a ductile material, the saw cut will branch blunt so that great effort is required to make it progress. However, brittleness and ductility are not inherent in the atoms that makes up a solid. In fact most solid have a definite temperature at which they make a transition from brittle to ductile behavior.

In a brittle material, the key ingredient turns out to be the crack dynamics. In fact, a crack that does not move is harmless and it has been observed that below a critical velocity the crack propagates smoothly; above the critical threshold the crack branches and zigzags leaving increasing rough surfaces. In any case the speed of the crack never reaches the Rayleigh speed of sounds as predicted by

simple argument but rather approach half of its value.

The idea of using our 2-dimensional sheet approach as a model of crack dynamics is to assume that a bound linking two connected atoms may break if the local deformation exceeds some given threshold. Even if our lattice is regular and cannot take into account dislocation processes, spatial disorder is present at different level: if the temperature $T \neq 0$ the position of the atoms fluctuate and a different breaking threshold may be possibly set for each bound. Once a bound is broken, the atoms on each side of the crack behave as free ends and their dynamics is governed by the introduction of virtual particles as explained in the previous section. A broken link weakens the material because the local deformation can no longer be distributed uniformly along the four neighbors.

Our model for fracture is a generalization of the sheet model allowing a different speed of sound at every lattice position. It is the exact interpretation of our LB wave model in the context of the solid body motion. Again, the dynamic alternates between two sub-lattices arranged as a checkerboard. However, the idea is that the particle do not jump to the exact symmetric position with respect to the center of gravity of its neighbors but only a fraction of it. This could be achieved by putting an attenuation factor in the expression (5.4), but if we want to conserve the local energy and momentum, a fifth quantities \mathbf{f}_0 must be introduced to store what was lost with the smaller jump. More precisely, if \mathbf{f}_α has the same meaning as before, the displacement $\mathbf{r}(i, j)$ of the moving particle was (in the coordinate frame attached to the particle)

$$\mathbf{r}(i', j') = 2\mathbf{r}_g(i', j') \doteq \frac{1}{2}(\mathbf{f}_1(i', j') + \mathbf{f}_2(i', j') + \mathbf{f}_3(i', j') + \mathbf{f}_4(i', j'))$$

and now becomes

$$\mathbf{r}(i', j') = \frac{1}{n^2} \left(2\mathbf{r}_g(i', j') + \frac{m_0}{2} \mathbf{f}_0 \right)$$

whereas

$$\mathbf{f}_0(i', j', t + 1) = m_0 \mathbf{r}(i', j') \Leftrightarrow \mathbf{f}_0(i', j')$$

where $n > 1$ is the indices of refraction and $m_0 = 2\sqrt{n^2} \Leftrightarrow 1$. This dynamics leads to a propagation matrix W for the \mathbf{f} 's which is equal to that given in (2.56). The mass of the particles is as before: 1 inside, 1/2 on the boundary and 1/4 on the corners; see (5.5) The momentum is given by

$$\mathbf{P} = \sum_{i', j'} m(i', j') \mathbf{r}(i', j') + \sum_{i'', j''} m(i'', j'') \frac{m_0}{2} \mathbf{f}_0(i'', j'')$$

where the $''$ indicates the particles that are not moving during the next time step whereas $'$ indicates as usual the moving particles. The interpretation is clear:

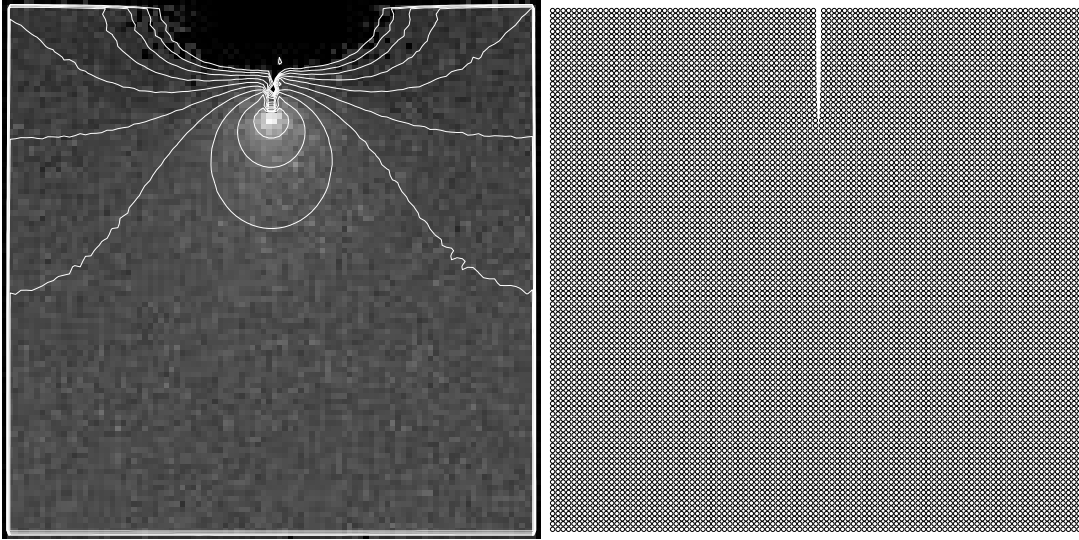


FIGURE 5.9: The figure represents (left) the contour plot of the energy E as defined in (5.7). The 100×100 square sheet is under a preloaded (mode I) stress ($S = 2.2 \times 10^{-2}$) but the fracture (initiated in the middle of the upper boundary: see the atoms position on the right) has stopped after few broken bounds because the noisy breaking threshold $\epsilon \approx 9 \times 10^{-3}$ was in that case too important. Thus we have a snapshot of the stationary stress pattern around the notch: the contour line mark the levels $\epsilon/E \in \{10, 20, 30, 40, 50, 60, 70, 80, 90, 100\}$.

part of the momentum is stored in the \mathbf{f}_0 and thus the sub-lattice at rest must be considered. The total energy E is given by

$$E = \frac{1}{2} \sum_{i,j} m(i,j) \left(\frac{1}{2} \mathbf{f}_0^2(i,j) + \frac{1}{4} \sum_{\alpha} \mathbf{f}_{\alpha}^2(i,j) \right)$$

In order to define the critical threshold ϵ above which a given link breaks, we shall write E in four distinct parts according to the four direction of the lattice:

$$E = \frac{1}{2} \sum_{i,j} \frac{m(i,j)}{4} (E_1(i,j) + E_2(i,j) + E_3(i,j) + E_4(i,j)) \quad (5.7)$$

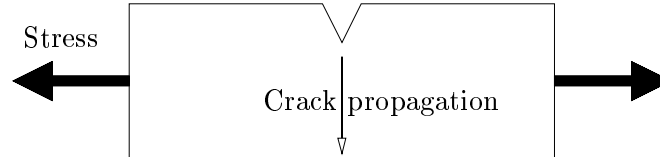
where

$$E_{\alpha} \doteq \mathbf{f}_{\alpha}^2 + \frac{1}{2} \mathbf{f}_0^2$$

Same experiments as in the previous section may be carried out to check the consistency of these definitions for a square sheet evolving in a rigid box, but this is not the purpose of this section.

The breaking rule

A typical experiment which is performed when studying fracture formation is to apply a stress by pulling in opposite way the left and right extremities of a solid sample. To achieve a static stress, the solid is prepared in a configuration where the x-spacing between the atoms is increased to the value $|r_0|(1 + S)$ where S is called the stress factor. Both left and right extremities are not allowed to move³. The initial temperature may be different from 0 and a small notch (artificially broken links) is made slowly in the middle of the sample to favor the apparition of the fracture at this position. In the fracture community, this is known as mode I loading, schematically we have



At each time step the evolution is performed normally and the black or white particles are moving according to the microscopic evolution rule. Before each move, the local energy E_α is calculated for each link and if the condition $E_\alpha > \epsilon$ is met, the link is broken and the dynamics will take into account an additional virtual particles to replace the broken link. The parameter ϵ is called the breaking threshold. On figure 5.9, the fracture did not propagate across the sample but stops after few steps giving rise to a stationary spatial distribution of stress around the notch.

On figure 5.10 we can see the result of two fracture experiments where the crack spontaneously propagates throughout the sample after being initiated artificially. An additional parameter μ was added, which turns out to be essential and eventually distinguishes the two cases presented here. The coefficient μ is a dissipation coefficient identical to what was introduced in (2.43). This attenuation is essential for preventing the reflection of too much energy from the boundaries toward the crack. With $\mu = 0.91$ the dissipation of energy is high enough to limit the acceleration of the crack up to some speed below the critical speed and the crack remains smooth. Whereas with $\mu = 0.96$ the crack accelerate above the critical speed and the instabilities appears: the crack progresses while making micro-branching. As can be seen on figure 5.10 the limiting speed is around half of the speed of sound in the sample.

The simulations were performed on samples with a speed of sound of about $1/\sqrt{2}$, but we could observe similar behaviors using another value for c . The critical speed of about $c/2$ is well captured in our model and this is a promising result. The model should be further analyzed and may provide new results using its available features like the possibility to have a temperature dependent thresh-

³This is a stress with constant distance; other experiments may be carried out to simulate a constant force

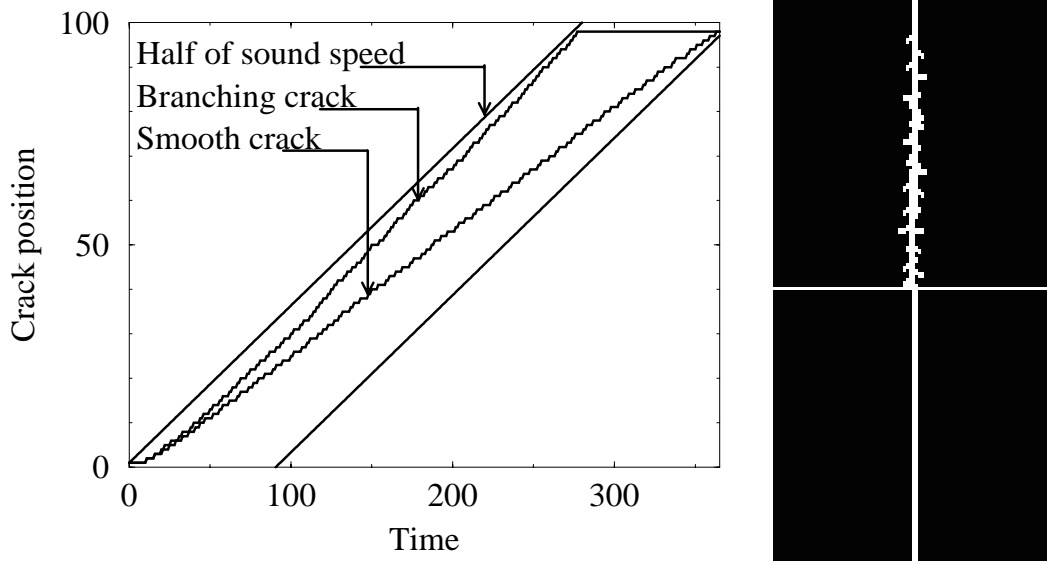


FIGURE 5.10: Two different types of fracture dynamics corresponding to the mode I loading. On the right is shown the position of the fracture edge in function of time (iterations) with the additional lines corresponding to half of the speed of sound to guide the eye. On the right is a snapshot of the broken bounds for the smooth branching crack (top: $S = 0.03$, $\mu = 0.96$, $\epsilon \approx 0.0058$) and the smooth crack (bottom: $S = 0.03$, $\mu = 0.91$, $\epsilon \approx 0.0058$).

old $\epsilon = \epsilon(T)$ or a temperature dependent speed of sound $n = n(T)$ for instance. As quantitative experimental results are now possible on a microscopical scale, such as for instance on the dynamics of the apparition of micro-cracks before the main crack, we expect our model will become an interesting tool to highlight the essential mechanisms of fracture.

5.2 Wave Localization

In this section, we study another different problem for which our model is useful. A coherent, but by no means complete, understanding of the difficult problem of waves in disordered media has only emerged recently[53]. Disordered media means here that the waves are supposed to undergo multiple scattering and the problem is quite different from the urban medium considered in chapter 4. The two problems are not logically separated⁴ but the treatment of radio waves

⁴The scattering of electromagnetic waves by systems whose individual dimensions are small compared with a wavelength is a common and important occurrence. In such interactions it is

propagation in urban microcell involves obstacles with typical sizes much larger than wave length. Thus, it is rather a diffraction problem, involving departures from geometrical optics caused by the finite wavelength of the waves and actual scattering analysis enters the game only once the small scale roughness of the buildings or the corners are taken into account.

For the scattering of waves by systems characteristic sizes are small compared to a wavelength, it is convenient to think of the incident fields as inducing a response that oscillates in definite phase relationship with the incident wave and radiates energy in directions other than the direction of incidence. If the medium contains randomly distributed such small scatterers, the picture of multiple scattered waves is very different from that we normally associate with waves. In particular the wave can become localized in a purely elastic medium, namely showing up no more spatial periodicity or possibility for transport.

The consequence of multiple wave scattering must be described according to the various natural length scales characterizing the problem. If \mathcal{L} , the wave length, is much larger than \mathcal{R} , the inter-scatterers distance, *i.e.* $\mathcal{L} \gg \mathcal{R}$, the scattering is weak and the medium appears as a homogeneous effective medium on the scale of few mean-free path. Roughly speaking, the mean-free path is the distance the wave can freely propagate between two scatterers. On the scale of many mean-free path, however, the waves end up by being completely randomized and show diffusive behavior like a classical Brownian particle does. If $\mathcal{L} \ll \mathcal{R}$ waves also show diffusive transport property but without the presence of an effective medium at any scale of observation since the waves can now resolve the microscopic disorder[53].

The fact that wave transport may be diffusive has a clear illustration in heat conduction. On the one hand it is known that heat conduction in solids is governed by the diffusion equation, and on the other hand, it is well known from statistical mechanics that heat in insulator is carried out by randomly scattered elastic waves. But the diffusion description of multiple scattered waves cannot be completely accurate since, whatever its appearance, the multiple scattered

convenient to think of the incident (radiation) fields as inducing electric and magnetic multi-poles that oscillate in definite phase relationship with the incident wave and radiate energy in directions other than the direction of incidence. If the wave length of the radiation is long compared to the size of the scatterer, only the lowest multi-poles (electric and magnetic dipoles) are important.

Although scattering and diffraction are not logically separate, the treatments tend to be separated, with diffraction being associated with departures from geometrical optics caused by the finite wavelength of the waves. Thus diffraction traditionally involves apertures or obstacles whose dimensions are large compared to a wavelength. To lowest approximation the interaction of electromagnetic waves is described by ray tracing (geometrical optics). The next approximation involves the diffraction of the waves around the obstacles or through the apertures with a consequent spreading of the waves. Simple arguments shows that the angle of deflection of the waves are confined to the region $\theta \leq \lambda/d$ where λ is the wavelength and d a linear dimension of the aperture or obstacle (approximations considered work well if $\lambda/d \ll 1$). Cited from [18]

wave is still a solution of the wave equation and therefore must satisfy the basic properties of its solutions. The basic phenomenon that makes wave diffusion original and different from classical diffusion is the *coherent backscattering effect*, or the weak-localization effect. This represents not only a deviation from classical diffusion but also the precursor to wave *localization*.

The coherent backscattering effect is the preservation of the phase coherence in the direction opposite to the incident direction during a scattering process. This is enforcing the backscattering through constructive interference and the medium show up a downward renormalized diffusion constant. The decrease in the diffusion constant is proportional to the scattering strength and if the diffusion constant eventually vanish we observe wave localization. The backscattering effect is fully operative only if the system is time-reversal invariant and if the scatterings are elastic. Moreover the negative correction to the diffusion constant is a monotonically increasing function of the physical sample size \mathcal{S} . On the one hand this is a striking conclusion because it makes the diffusion constant no longer an intensive quantity, but on the other hand this a natural consequence of the additional length scale inevitably introduces by a localization process: the localization length.

To understand the actual role of the coherent backscattering for the localization phenomenon, it is interesting to point out the analogy between a random walker and a multiple scattered wave. The classical random walk, or Brownian motion, is not time reversible and a convenient picture of the process is to imagine a test particle having a ballistic motion between two elastic collisions with the bulk particles constituting the medium. During a collision process the test particle is “scattered”, *i.e.* the direction of its speed is changed in a completely random manner (uncorrelated in time and space). The well known diffusion behavior of this particle concerns the probability of finding it at some time and at a certain distance away from the origin; it will show up only after **a configuration averaging**. Now, suppose the random walk is made time-reversal invariant: for instance, the scattering centers, or bulk particles, are immobile and quenched once for ever. For this so-called Lorentz gas, if a particle eventually bounce back, it will then exactly trace its own way back to the origin. This will considerably reduce the space the particle is visiting and the diffusion constant shown up after averaging over an ensemble of these frozen scatterer configurations is downward renormalized [54].

The coherent back-scattering effect is the wave analogous of this situation. By using light scattering from fixed scatterer configuration, it was found that for each configuration, there were many intensity peaks occurring at arbitrary reflection angles. This is a so called speckle pattern⁵. However, after configurational

⁵Speckle patterns are commonly observed: for instance if you illuminate a non-perfect surface by a coherent laser beam and project the reflected light on a sheet of paper, you will find interlaced bright and dark spot well visible; these intensity variation are called the speckle pattern.

averaging only the backscattering peak (the peak in the direction of the source) emerges, and all the other peaks vanishes. This is because, on each specific path, the successive portions of the traveling wave that are tracing their own way back interfere additively.

From the Lorentz gas analogy it is clear that the downward renormalization of the diffusion constant for truly time reversal invariant medium, is a monotonically increasing function of the sample size. To the increase of the sample size corresponds an increase of the probability to an eventually bounce back on a scatterer. Another striking property of this downward diffusion renormalization, or weak localization effect, is its dependence in the spatial dimensionality of the sample. This again may be anticipated considering the criticality of the dimension 2 as far as the diffusion process is concerned: after a time t the particle visits a “surface” around the origin which turns out to be dense in 1- and 2-dimensional space but sparse in 3-dimensional space. Finally one finds [53] that the negative correction to the diffusion constant D for wave diffusion have the approximative following form

$$\Delta_D = \begin{cases} \frac{1}{\mathcal{R}} \Leftrightarrow \frac{1}{\mathcal{S}} & \text{3-dim} \\ \ln \frac{\mathcal{S}}{\mathcal{R}} & \text{2-dim} \\ \mathcal{S} \Leftrightarrow \mathcal{R} & \text{1-dim} \end{cases}$$

where \mathcal{R} is the mean free path and \mathcal{S} the sample size. One immediately sees that the localization tendency, namely the opportunity for Δ_D to diverge, corresponding to $D \rightarrow 0$, prevails in 1- and 2-dimensional space whereas the possibility of 3-dimensional classical wave localization remains an open question.

Our model for wave propagation developed in chapter 2 is particularly well adapted to numerically investigate wave propagation of a scalar quantity, I in random media beyond what is analytically possible. The main questions that may be addressed by our approach concern (i) the analysis of the (smooth) transition regimes between the effective medium, the diffusive, and the localization behavior and (ii) the analysis of the dynamics of the propagation on some particular configuration before averaging.

On a square lattice, I is defined as a linear combination of four traveling flux f_i (one per lattice direction as labeled in figure 2.1 and a rest flux f_0):

$$I = 2\sqrt{n^2 \Leftrightarrow 1}f_0 + f_1 + f_2 + f_3 + f_4$$

the collision and the streaming process undergone by the f 's is given by mean of a matrix W , as in equation (2.41). With the definition of I given above and the definition (2.40) of the refraction index n , W may take the symmetric form (A.5) where, on purpose, we retain ξ , the relaxation time of the LBGK dynamics as a free parameter.

Our 2-dimensional media is defined by mean of two different index of refraction n : the background have a value of $n_0 = 1$ and the randomly distributed scatterers

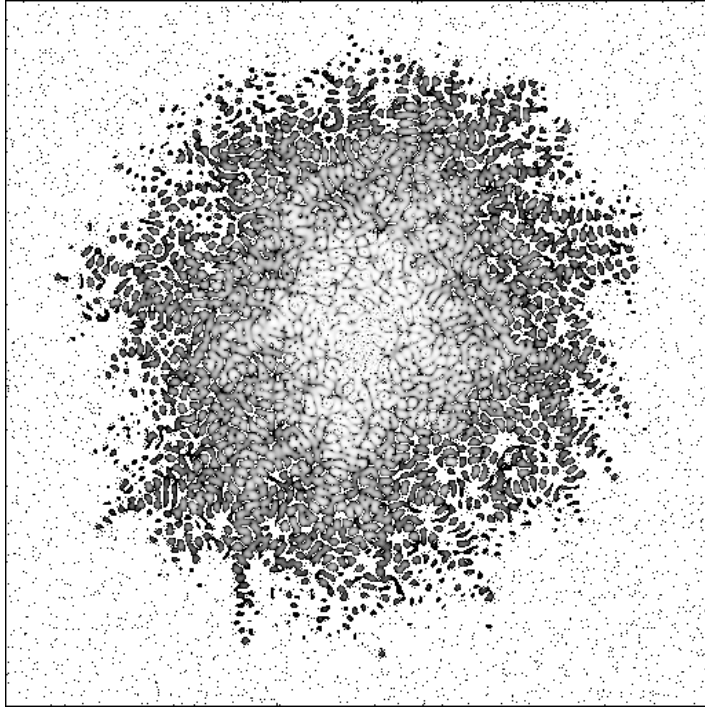


FIGURE 5.11: *Snapshot of energy propagation pattern in a random medium composed of a background with the index of refraction $n_0 = 1$ containing 2% of randomly distributed scatterers (black dots) all with $n_1 = 10$. The whole sample has $\xi = 1/2$ so that the dynamics is time reversal invariant. The source is placed in the center of the sample and oscillates with a period of 16 time steps. The pattern shows large fluctuations (a speckle pattern) and the diffusive, or sub-diffusive behavior only emerges after an averaging over different such configurations.*

have a value of $n_1 > 1$. Note that different medium may be designed, for instance we could have chosen a different value of n for each scatterer or even a different value of n for each lattice site. Figure 5.11 show the typical aspect of the energy propagation pattern issued from a centered point source in a random medium composed of 2% of scatterers. The pattern shows large fluctuations and further analysis or comparison with classical diffusion will involve some kind of averaging over different spatial configurations.

To avoid the excess of computation generated by an averaging process over successive configuration, we choose a 2-dimensional system with a 1-dimensional symmetry. Thus, the averaging is achieved by a reduction along the “irrelevant” dimension. The purpose is to determine the nature of the propagation of the energy issued from a “line-pulse” in a two-dimensional long strip-like medium (typically of size 4096×64). The line source is placed in the middle, and it is oriented parallel to the width of the strip. It radiates synchronously two oscilla-

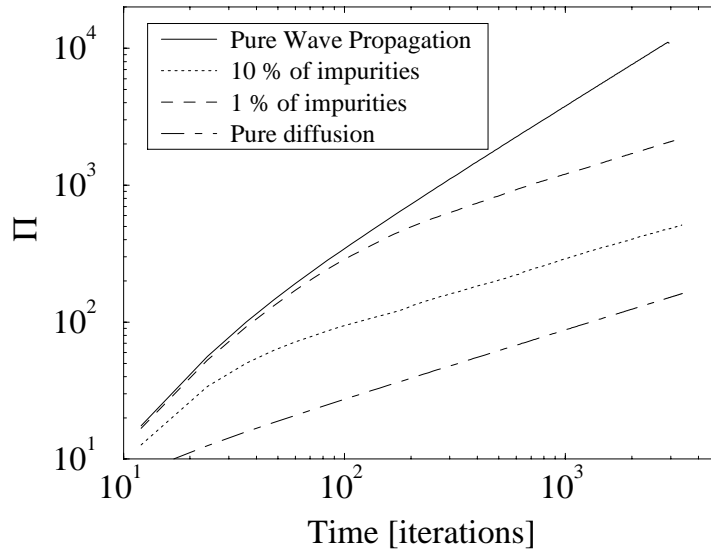


FIGURE 5.12: *Transition from the wave- to the diffusive transport in a 1-dimensional geometry. The strip-like domain size is 4096×64 and the refraction index ratio of background over impurities is $1/10$. The square root of the second moment of the energy distribution Π is plotted in function of time. Between the two extreme cases: homogeneous medium and pure diffusion we observe a smooth transition for random media (1% and 10%) from the wave regime $\propto t$ to the diffusive regime $\propto \sqrt{t}$.*

tions of a wave with a given period $T = 6$. Two free parameters are completely fixing the medium: ρ the density of randomly distributed scatterers and n_1 the homogeneous scattering strength, or refraction index of the scatterers. If A is the local value of the amplitude of the wave, we measure the dynamics of

$$\Pi(t) = \sqrt{\int A^2(\vec{r}, t) r^2 d^2 r}$$

where r is the distance to the line source and A is given by (4.1). Thus Π is the square root of the second moment of the energy distribution.

The results are shown on figure 5.12. It can be seen that for the homogeneous medium, a pure wave propagation is characterized, as expected, by $\Pi(t) \propto t$. For random non-dissipative media the dynamics switches from an initial regime equivalent to the initial regime of the homogeneous case to a behavior given by $\Pi(t) \propto \sqrt{t}$ which is typical of a diffusive transport regime. The cross-over is smooth and happens earlier in case of increasing disorder (or increasing scattering strength). Our model allows us to simulate true diffusion for comparison with only a slight change in the propagation matrix. Indeed, it has been established in [30] that choosing $W = \mathbf{1}/4$ allows us to simulate the diffusion equation with a lattice

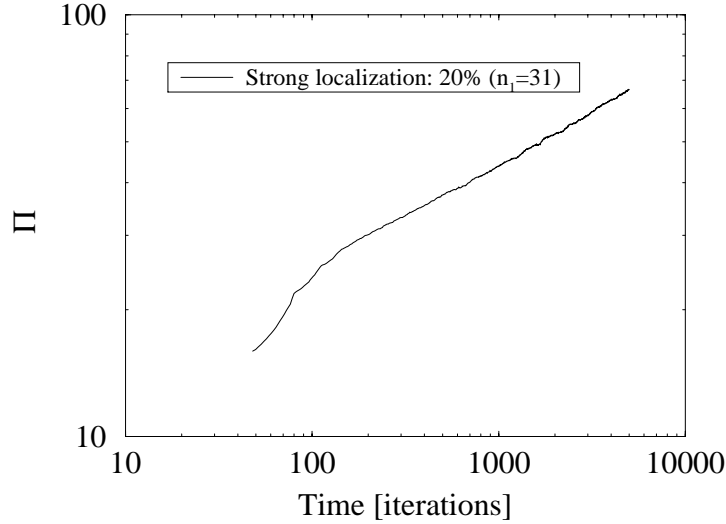


FIGURE 5.13: The layout is the same as in figure 5.12 but the disorder is much bigger: $n_1 = 30$ with an impurity density of 20%. The regime is clearly sub-diffusive or localizing because we have $\Pi \propto t^\alpha$ with α clearly smaller than $1/2$.

Boltzmann like system. For the diffusion case, Π must be changed: instead of the “energy” A^2 we take the local field value $I = \sum_i m_i f_i$; and the good agreement between classical diffusion and wave diffusion (or weak localization) is shown in 5.12.

Strong localization is presented in [53] as a tendency for the diffusion coefficient to fall towards 0. We believe a better understanding is given in figure 5.13 where the strong localization effect is characterized by a behavior $\Pi \propto t^\alpha$ where $\alpha < 1/2$. Thus, we could not assess a distance above which the wave is strictly unable to penetrate (true localization) but this may be likely to come from the finite size of our sample and we believe that eventually $\alpha \rightarrow 0$ at a larger scale of observation.

It is interesting to note that, in our strong localization limit (large ρ and large n), the scatterers become true reflectors. However our system do not dissipate energy. Now, if we think to replace the configurational averaging by a spatial averaging (to reduce computational time, for instance) we get a propagation model which is a superposition of free propagation (W) and reflection ($\Leftrightarrow R$):

$$(1 \Leftrightarrow \rho)W \Leftrightarrow \rho R$$

which is exactly the dissipation matrix proposed in (2.44). Finally, this is demonstrating once more that there is different levels of reality and one cannot go innocently from one to another.

The purpose of the past two sections was not to predict new physical results. The purpose was rather prospective, namely to present alternative applications

of our model and show how easy it is to adapt it to a completely new class of problems. The promising perspectives were assessed by exhibition of relatively well known - but only recently obtained - phenomena naturally emerging out of our model. These properties are not straightforward and are still subjects to active researches. We did not plug in our model any *ad hoc* properties, this demonstrates the kind of universality caught by our model. This an important point towards a global assessments of the approach. We believe the study of fracture phenomena or of the wave localization by means of a lattice Boltzmann approach is still in a very early stage and is worth further studies.

Conclusion

The microscopic simulation approach to complexity out-does any other approach because it naturally takes into account the various levels of reality. It has been shown that for many (however not for all) problems, simulation provides new solutions beyond the analytical approach using differential equations. For computational physicist, performing numerous simulations in order to explore a vast parameter space is often the key to new findings or new understandings. Thus, it is of first priority to run fast processes and the growing available computing resources must be optimally used, in particular, all parallelism opportunities must be cleverly exploited.

Complexity and the apparent hierarchy of reality levels is not the appanage of the physical, inert nature. On the contrary, emerging collective behaviors out of local and “atomic” interactions is quite an universal property of non-trivial systems and may be exhibited in a wide range of phenomena. Living cells, insects, economic actors, car drivers or citizens may eventually be compared to simple “particles” if the ensemble is studied at a larger length and time scale. This makes the simulation approach general and allow us to study historically separated problems in a common framework. This new trend is already a progress on its own because it opens the scientist’s mind and ensure efficient interdisciplinary progress. Among the various ways to make simulation, lattice modeling technics are even more challenging because they are forced to mimic the underlining microscopy in the most simple and abstract manner: space and time being *ab initio* fully discrete.

One of the least comfortable (but also one of the least unexpected) aspect of this new trend is actually the resistance it provokes among a class of rather

conservative scientists. It is well known that nothing is more difficult than to jeopardize set ideas and virtual “atoms” or discrete ingredients are often believed to be too simple to have the slightest chance to give a correct answer to any relevant question. In the scientific community, the change induced by the use of a model, say a language, that bypass the usual differential equation apparatus, is still hardly accepted. Starting with simple ingredients is surprisingly hard for those people who are too much aware of the apparent global complexity. A common attitude that we faced while promoting the results of our models in the community of engineers, was an excessive focus and doubt on some details that may be rightly considered as wrong with respect to some present knowledge but that we believe and eventually show to be irrelevant in the discussion. If we were less modest, we would dare to answer by Matthew 7:3 “Why do you look at the speck of sawdust in your brother’s eye and pay no attention to the plank in your own eye?”

The saga about the “wrong” wavelength we used in our radio wave propagation simulations is a good example of the above mentioned difficulties. As explained in sub-section 4.3, the wavelength of our automata measured in the discretized layout is significantly bigger than the corresponding real wavelength used in the measurements. For the radio science community it was hardly acceptable simplification as the frequency is the most important parameter of their predictions and measurements. But we showed, with our renormalization method, how to correct the most important errors introduced by our choice. Finally, we believe our wavelength turns out to be adapted to the disorder of our discretized layout for eventually giving a good agreement between measurements and predictions.

However, “*comparaison n’est pas raison*” and computational physicists must avoid too much enthusiasm for their discrete models. The design of a new model is an exiting and very creative exercise but the freedom is so large that almost anything may be implemented. Consequently, the necessary effort to gain confidence in the outcome of a simulation is hard. The most difficult step is to transform something which looks nice in something that explains much. For this point the key is to ask the right question, and the expected increase of computational power, as formidable as it can be, will not help much for that step.

We believe the perspective of the present work remains open and much must be said before having the last word. Our model for radio wave propagation is now one of the corner stones of an ongoing research and development project focusing on the difficult optimization problem to distribute fixed antennas in mobile communication environment. Our model turns out to be suitable for urban environment. Like everything, a model must evolve if it is to survive in the scientific and the engineer community; for instance, it would be straightforward to generalize our model for 3-dimensional layout. But this may be not the right

solution if one establishes that a finer 2-dimensional layout (using a realistic wavelength) is a better way to use futur increasing computational power. As for the fracture dynamics problem and the wave localization phenomena the open questions are numerous and still of fundamental interest: what bounds crack velocity? what makes a material brittle or ductile? do waves localize in 3-dimensional samples ? Thus 3-dimensional simulations together with the analysis of new types of measurements are expected to provide significant progress in a near future.

The Matrix Warehouse

This appendix recalls the various matrices encountered throughout the presentation of the lattice Boltzmann method for wave propagation. Some of them are essentially equivalent but a different accent is given by the choice of the parameters involved. The notations and the directions labeling correspond to figure 2.1.

A.1 The TLM-matrix

This the most simple wave propagation matrix on a $4\mathcal{N}2$ lattice. The simulated index of refraction is fixed $n = 1.0$ corresponding to a wave speed $c = v/\sqrt{2}$:

$$\frac{1}{2} \begin{pmatrix} 1 & 1 & \Leftrightarrow 1 & 1 \\ 1 & 1 & 1 & \Leftrightarrow 1 \\ \Leftrightarrow 1 & 1 & 1 & 1 \\ 1 & \Leftrightarrow 1 & 1 & 1 \end{pmatrix} \quad (\text{A.1})$$

A.2 The *Parflow*-matrix or the (c_1, c_2) -matrix

This matrix is used in the kernel of the application *Parflow*. It corresponds to a generalization of the TLM matrix with the introduction of two coefficient allowing to define absorbing as well as reflecting sites:

$$c_1 \begin{pmatrix} c_2 & c_2 & c_2 \Leftrightarrow 1 & c_2 \\ c_2 & c_2 & c_2 & c_2 \Leftrightarrow 1 \\ c_2 \Leftrightarrow 1 & c_2 & c_2 & c_2 \\ c_2 & c_2 \Leftrightarrow 1 & c_2 & c_2 \end{pmatrix} \quad (\text{A.2})$$

For the case $(c_1, c_2) = (1, 1/2)$ we obtain matrix A.1.

A.3 The asymmetric (a, N, \mathcal{N}) -matrix

This matrix is the most straightforward matrix obtained from the mathematical derivation presented in the chapter 2. It allows an easy generalization for any lattices opposite velocity bounds (any dimension \mathcal{N}): N is the coordination number ($N + 1$ fields are needed to take into account rest particles) and a enters the equilibrium distribution (see 2.9). For the case $4\mathcal{N}2$ we obtain

$$W = \frac{2a}{N+1} \cdot \begin{pmatrix} \frac{N+1}{2a} - N & \frac{N+1}{a} - N & \frac{N+1}{a} - N & \frac{N+1}{a} - N & \frac{N+1}{a} - N \\ 1 & 1 - \frac{N+1}{a}(\frac{1}{2} - \frac{\mathcal{N}}{N}) & 1 & 1 - \frac{N+1}{a}(\frac{1}{2} - \frac{\mathcal{N}}{N}) & 1 \\ 1 & 1 & 1 - \frac{N+1}{a}(\frac{1}{2} - \frac{\mathcal{N}}{N}) & 1 & 1 - \frac{N+1}{a}(\frac{1}{2} - \frac{\mathcal{N}}{N}) \\ 1 & 1 - \frac{N+1}{a}(\frac{1}{2} - \frac{\mathcal{N}}{N}) & 1 & 1 - \frac{N+1}{a}(\frac{1}{2} - \frac{\mathcal{N}}{N}) & 1 \\ 1 & 1 & 1 - \frac{N+1}{a}(\frac{1}{2} - \frac{\mathcal{N}}{N}) & 1 & 1 - \frac{N+1}{a}(\frac{1}{2} - \frac{\mathcal{N}}{N}) \end{pmatrix} \quad (\text{A.3})$$

This corresponds to a wave propagation speed c and a index of refraction given by

$$c = \sqrt{\frac{a}{\mathcal{N}} \frac{N}{N+1}} \text{ and } n = \sqrt{\frac{N+1}{aN}}$$

A.4 The symmetric (n) -matrix

This matrix is obtained after a change of variable which is equivalent to have a different mass ($m_0 \neq 1$) for the rest particles (depending on the index of refraction) in the definition of the $I = \sum_i m_i f_i$. We must set $m_0 = 2\sqrt{n^2} \Leftrightarrow 1$, $m_{i \neq 0} = 1$ and we obtain ($c = 1/n\sqrt{2}$):

$$\frac{1}{2n^2} \begin{pmatrix} 2n^2 \Leftrightarrow 4 & 2\sqrt{n^2} \Leftrightarrow 1 & 2\sqrt{n^2} \Leftrightarrow 1 & 2\sqrt{n^2} \Leftrightarrow 1 & 2\sqrt{n^2} \Leftrightarrow 1 \\ 2\sqrt{n^2} \Leftrightarrow 1 & 1 & 1 & 1 \Leftrightarrow 2n^2 & 1 \\ 2\sqrt{n^2} \Leftrightarrow 1 & 1 & 1 & 1 & 1 \Leftrightarrow 2n^2 \\ 2\sqrt{n^2} \Leftrightarrow 1 & 1 \Leftrightarrow 2n^2 & 1 & 1 & 1 \\ 2\sqrt{n^2} \Leftrightarrow 1 & 1 & 1 \Leftrightarrow 2n^2 & 1 & 1 \end{pmatrix} \quad (\text{A.4})$$

A.5 The symmetric (n, ξ) -matrix

This is the same matrix as before, but we have explicitly kept the relaxation time ξ . The non-dissipative (or non-viscous) case is obtained with $\xi = 0.5$:

$$\frac{1}{\xi 4n^2} \begin{pmatrix} 4(\xi n^2 \Leftrightarrow 1) & 2\sqrt{n^2 \Leftrightarrow 1} & 2\sqrt{n^2 \Leftrightarrow 1} & 2\sqrt{n^2 \Leftrightarrow 1} & 2\sqrt{n^2 \Leftrightarrow 1} \\ 2\sqrt{n^2 \Leftrightarrow 1} & 1 + 4\xi \Leftrightarrow 2 & 1 & 1 \Leftrightarrow 2n^2 & 1 \\ 2\sqrt{n^2 \Leftrightarrow 1} & 1 & 1 + 4\xi \Leftrightarrow 2 & 1 & 1 \Leftrightarrow 2n^2 \\ 2\sqrt{n^2 \Leftrightarrow 1} & 1 \Leftrightarrow 2n^2 & 1 & 1 + 4\xi \Leftrightarrow 2 & 1 \\ 2\sqrt{n^2 \Leftrightarrow 1} & 1 & 1 \Leftrightarrow 2n^2 & 1 & 1 + 4\xi \Leftrightarrow 2 \end{pmatrix} \quad (\text{A.5})$$

A.6 The 1-dimensional matrix

The following matrix is for the special 1-dimensional case of a wave traveling a speed c :

$$\begin{pmatrix} 1 \Leftrightarrow 2c^2 & 2 \Leftrightarrow 2c^2 & 2 \Leftrightarrow 2c^2 \\ c^2 & c^2 & c^2 \Leftrightarrow 1 \\ c^2 & c^2 \Leftrightarrow 1 & c^2 \end{pmatrix} \quad (\text{A.6})$$

A.7 The diffusion matrix

In [30] is derived a lattice Boltzmann method for simulating the diffusion equation. The propagation matrix is very similar to the wave case (basically there is a sign change) and is given here for comparison:

$$\begin{pmatrix} p_0 & p & p_2 & p \\ p & p_0 & p & p_2 \\ p_2 & p & p_0 & p \\ p & p_2 & p & p_0 \end{pmatrix} \quad (\text{A.7})$$

If $p_0 + 2p + p_2 = 1$ we simulate a diffusion coefficient D given in the lattice units:

$$D = \frac{p + p_0}{4(1 \Leftrightarrow (p + p_0))}$$

The PVM Code of the Model

This appendix presents the complete C code of the benchmark used in the performance analysis throughout the chapter 3. The version given here contain the calls to PVM routines to run the code in parallel. The PVM calls connect the various sub-domains by exchanging the value of the flux crossing the sub-domain boundaries. If the code is to be run on one single sequential machine, then some rewriting is necessary to skip the PVM calls and merge the computation of the boundaries with the computation inside the domain.

The most important part is of course the kernel of the application to be found in the subroutine called `void TLM()`. The reason to give here an exhaustive, and readily compilable code is that the performances obtained on the various platforms may dramatically depend on small details that must be fixed once for all to avoid confusion.

```

/*****
/*      L A T T I C E   B O L T Z M A N N   W A V E   A U T O M A T A           */
/*                                                                 */
/*                                                                 */
/*          in C, using the PVM library for parallelization        */
/*          by Pascal O. Luthi, University of Geneva                */
/*          October 1997                                           */
/*                                                                 */
/*          COMPILATION for Sunstations at CUI:                    */
/*                                                                 */
/*      cc -fast tlmPVM.c                                          */
/*          -I/unige/pvm3/include                                  */
/*          -L/unige/pvm3/lib/SUN4SOL2                            */
/*          -lgpvm3                                               */
/*          -lpvm3                                                */

```

```

/*      -lm                                                    */
/*      -lsocket                                              */
/*      -lnsl                                                 */
/*      -lthread                                              */
/*      -lX11                                                 */
/*      */
/*      INVOQUE the executable with 2 arguments:              */
/*      */
/*      "a.out 600 12" = a 600 X 600 array on 12 processors  */
/*      */
/*****

/*****
/* INCLUDES                                                    */
/*****

#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include <time.h>
#include <pvm3.h>
#include <sys/types.h>

/*****
/* DEFINITION                                                    */
/*****

/***** Defintion of the discrete period of the source          *****/
#define PERIOD 8

/***** Defintion a amplitude for generating the source          *****/
#define GAMMA 0.07015

/***** Definition of Pi                                          *****/
#define PI 3.141592654

/***** Definition of absorbing coefficient on the border of the domain *****/
#define C0 0.0
#define C1 0.436
#define C2 0.8

/***** Definition of special tags used with PVM communication routines *****/
#define F4 1
#define F2 2
#define END 3
#define TIM 4

/***** Definition of special environement variables              *****/
#define dim env[0] /* Dimension of the domain                   *****/
#define nprocs env[1] /* Number of processors involved in the run *****/

```



```

/***** Amplitude of the source *****/
float amplitude;

/***** Number of byte's shift for propagation pahse *****/
int bigshift,smallshift;

/***** Flag if process contains the source *****/
int sourcei;

/***** Zero (sic) *****/
int zero=0;

/***** To integrate the density for extracting oscillation amplitude *****/
static float *_integral,**integral;

/***** To measure the delay between emission and reception *****/
static int *_delay,**delay;

/*****
/* MAIN PROGRAM */
/*****

main(int argc, char **argv)
{

/***** Intern variables returned or used with PVM calls *****/
    int pid,info,shutdown=0;
    int atid,atag,alen;
    int nchild;

/***** Loop ancilla integer *****/
    int i;

/***** Timming variables *****/
    long *tim1,*tim2;
    long t01,t11,mt1=0,t02,t12,mt2=0;

/***** Introduction, typically first PVM calls *****/
    tid = pvm_mytid();
    pid = pvm_parent();

/***** If I am the MASTER process (ie I have no parent) *****/
/*****
    if (pid==PvmNoParent){

/***** Read arguments (2) from command line *****/
    if(argc==3){

```

```

/***** Dimension of the global domain *****/
    dim      =(int)strtol(argv[1],NULL,10);

/***** Number of processors to work with *****/
    nprocs   =(int)strtol(argv[2],NULL,10);

/***** Some comments *****/
    printf("# TLM on arrays: %i X %i \n",dim,dim);
    printf("# with %i processors involved \n",nprocs);
    printf("# Iterations number: %i ... \n",dim);
}

/***** Wrong input format *****/
    if((argc!=3)|| (dim==0)|| (nprocs==0)){
        printf("# Error: wrong dimension or nprocs !\n");
        printf("# Call the executable with 2 integer arguments \n");
        printf("# dimension of the array and the number of procs\n");
        printf("# > a.out 100 4\n");
        exit(0);
    }

/***** The array that will contain the ID's is allocated *****/
    ids = (int*) malloc(nprocs*sizeof(int));

/***** Spawn the child process *****/
    if(nprocs>1) nchild=
        pvm_spawn(argv[0],NULL,PvmTaskDefault,"cuisun1",nprocs-1,ids);
    else nchild =0;
    printf("# %i Child processes spawn by master\n",nchild);

/***** Error message *****/
    if(nchild!=nprocs-1){
        printf("# Error: not all requested childs could be spawn\n");
        printf("# Exiting without killing survival slaves... \n");
        exit(0);
    }

/***** Last ID is master's ID *****/
    ids[nprocs-1]=tid;

/***** Calculate the process number (is not process ID) *****/
    tno = nprocs-1;

/***** Send the content of command line (dim and nprocs) because *****/
/***** not possible to be passed by the second arguement of spawn *****/
    for(i=0;i<nchild;i++)info=pvm_psend(ids[i],0,env,3,PVM_INT);

/***** Send the list of the task ID for later definition of neighbors *****/
    for(i=0;i<nchild;i++)info=pvm_psend(ids[i],i,ids,nprocs,PVM_INT);
}

```

```

/***** If I am not the MASTER process (I am a slave) *****/
/*****
else{

/***** Receive the content of command line (dim and nprocs) because *****/
/***** not possible to be passed by the second argument of spawn *****/
        info=pvm_precv(pid,-1,env,3,PVM_INT,&atid,&atag,&alen);

/***** Receive the list of the task ID for later definition of neighbors *****/
        ids = (int*) malloc(nprocs*sizeof(int));
        info=pvm_precv(pid,-1,ids,nprocs,PVM_INT,&atid,&atag,&alen);

/***** My own process number (ie is not my task ID) *****/
        tno = atag;
    }

/***** Everybody is doing the main calculation *****/
/*****

/***** As much iterations as the dimension extent *****/
        loop      = dim;

/***** The global array is beeing split over the processors involved *****/
        small_dim = dim/nprocs;

/***** If the source is not contained in my domain sourcei=-1 *****/
        sourcei   = (tno==(int)nprocs/2)?(dim/2)%small_dim:-1;

/***** Some comments *****/
        printf("# Local dimension %i %i \n",small_dim,dim);
        printf("# Source on procs %i at line %i \n",
                (int)(nprocs)/2,(dim/2)%small_dim);

/***** Determine neighborhood in terms of process number *****/
        tle      = (tno-1<0)      ?nprocs-1:(tno-1);
        tri      = (tno+1==nprocs)?0      :(tno+1);

/***** Source variables *****/
        period   = PERIOD;
        amplitude = GAMMA;

/***** Number of byte's shift for propagation pahse *****/
        bigshift = (dim*small_dim-1)*sizeof(float);
        smallshift = (small_dim-1)*sizeof(float *);

/***** Initialisation, allocation and construction of arrays *****/
        INIT();

```

```

/*****
/* MAIN LOOP *****/
/*****

    t02 = clock();
    t01 = second();

    for(i=0;i<loop;i++) TLM();

    t12 = clock()-t02;
    t11 = second()-t01;

/*****

/***** The End of the program *****/
    if (pid==PvmNoParent){

/***** If I am the MASTER process (ie I have no parent) *****/
        tim1 = (long*) malloc(nprocs*sizeof(long));
        tim2 = (long*) malloc(nprocs*sizeof(long));
        printf("# Master Elapsed time: %i, CPU %3.2f s\n",
                t11,t12/(float)CLOCKS_PER_SEC);

        tim1[nprocs-1] = t11;
        tim2[nprocs-1] = t12;

/***** Receive elapsed from slaves *****/
        for(i=0;i<nchild;i++)
            info=pvm_prevc(-1,TIM ,&tim1[i],1,PVM_LONG,&atid,&atag,&alen);
        for(i=0;i<nchild;i++)
            info=pvm_prevc(-1,TIM+1,&tim2[i],1,PVM_LONG,&atid,&atag,&alen);

/***** Print Slave results *****/
        for(i=0;i<nchild;i++)
            printf("# Child %i Elapsed time: %i, CPU %3.2f s\n",
                    i,tim1[i],tim2[i]/(float)CLOCKS_PER_SEC);

/***** Send Shutdown Signal *****/
        for(i=0;i<nchild;i++)info=pvm_psend(ids[i],END,NULL,1,PVM_INT);
        for(i=0;i<nchild;i++) while(pvm_pstat(ids[i])!=PvmNoTask);
    }
    else{

/***** If I am not the MASTER process (I am a slave) *****/

/***** Send result to master *****/
        info=pvm_psend(pid,TIM ,&t11,1,PVM_LONG);
        info=pvm_psend(pid,TIM+1,&t12,1,PVM_LONG);

```

```

/***** Recv the shutdown Signal *****/
    info=pvm_precv(pid,END,&shutdown,1,PVM_INT,&atid,&atag,&alen);
}
}

/***** THE KERNEL OF THE APPLICATION *****/
void TLM()
{

/***** Loop ancilla integer *****/
    register int i,j;

/***** Lattice boltzmann model variables *****/
    register float coef,f1,f2,f3,f4;
    register float *pt,psi;

/***** PVM variable *****/
    register int info;
    int atid,atag,alen;

/***** Time incrementation *****/
    tlm_time++;

/***** line number 0 *****/
/***** The collision process *****/
/***** *****/
    for (j=0;j<dim;j++)
        if(f.f1[0][j]!=0.0 || f.f2[0][j]!=0.0 ||
            f.f3[0][j]!=0.0 || f.f4[0][j]!=0.0 ){

            f1 = f.f1[0][j];
            f2 = f.f2[0][j];
            f3 = f.f3[0][j];
            f4 = f.f4[0][j];

            delay[0][j]++;
            psi=(f1+f2+f3+f4);

            integral[0][j]+=psi*psi;
            psi*=coeff.c2[0][j];
            coef=coeff.c1[0][j];

            f.f1[0][j]=coef*(psi-f3);
            f.f2[0][j]=coef*(psi-f4);
            f.f3[0][j]=coef*(psi-f1);
            f.f4[0][j]=coef*(psi-f2);
        }
}

```

```

    }

/***** Source *****/
    if(sourcei==0)
        f.f1[0][dim/2]=f.f2[0][dim/2]=
        f.f3[0][dim/2]=f.f4[0][dim/2]=
            amplitude*sin((double)(t1m_time)*PI*2.0/period);

/***** even the extrem boundary are sent *****/
    info=pvm_psend(ids[t1e],F4,f.f4[0],dim,PVM_FLOAT);

/***** line number small_dim-1 *****/
/***** The collision process *****/
/***** *****/
    for (j=0;j<dim;j++)
        if(f.f1[small_dim-1][j]!=0.0 || f.f2[small_dim-1][j]!=0.0 ||
            f.f3[small_dim-1][j]!=0.0 || f.f4[small_dim-1][j]!=0.0 ){

            f1 = f.f1[small_dim-1][j];
            f2 = f.f2[small_dim-1][j];
            f3 = f.f3[small_dim-1][j];
            f4 = f.f4[small_dim-1][j];

            delay[small_dim-1][j]++;
            psi=(f1+f2+f3+f4);

            integral[small_dim-1][j]+=psi*psi;
            psi*=coeff.c2[small_dim-1][j];
            coef=coeff.c1[small_dim-1][j];

            f.f1[small_dim-1][j]=coef*(psi-f3);
            f.f2[small_dim-1][j]=coef*(psi-f4);
            f.f3[small_dim-1][j]=coef*(psi-f1);
            f.f4[small_dim-1][j]=coef*(psi-f2);
        }

/***** even the extrem boundary are sent *****/
    info=pvm_psend(ids[tri],F2,f.f2[small_dim-1],dim,PVM_FLOAT);

/***** lines inside the domain *****/
/***** The collision process *****/
/***** *****/
    for (i=1;i<small_dim-1;i++)
        for (j=0;j<dim;j++)
            if(f.f1[i][j]!=0.0 || f.f2[i][j]!=0.0 ||
                f.f3[i][j]!=0.0 || f.f4[i][j]!=0.0 ){

                f1 = f.f1[i][j];
                f2 = f.f2[i][j];

```

```

        f3 = f.f3[i][j];
        f4 = f.f4[i][j];

        delay[i][j]++;
        psi=(f1+f2+f3+f4);

        integral[i][j]+=psi*psi;
        psi*=coeff.c2[i][j];
        coef=coeff.c1[i][j];

        f.f1[i][j]=coef*(psi-f3);
        f.f2[i][j]=coef*(psi-f4);
        f.f3[i][j]=coef*(psi-f1);
        f.f4[i][j]=coef*(psi-f2);
    }

/***** Source *****/
    if(sourcei>0)
        f.f1[sourcei][dim/2]=f.f2[sourcei][dim/2]=
        f.f3[sourcei][dim/2]=f.f4[sourcei][dim/2]=
        amplitude*sin((double)(t1m_time)*PI*2.0/period);

/***** The shift; spreading phase *****/
/*****
    pt=f.f2[small_dim-1];
    memmove(f.f2+1,f.f2,smallshift);
    f.f2[0]=pt;

    pt=f.f4[0];
    memmove(f.f4,f.f4+1,smallshift);
    f.f4[small_dim-1]=pt;

/***** Column shift with memmove: Here the adsorbing boundary *****/
/***** conditions with the last layer to 0 is necessary. Because *****/
/***** the shift is not really periodic. *****/
    memmove(f.f3[0],f.f3[0]+1,bigshift);
    memmove(f.f1[0]+1,f.f1[0],bigshift);

/***** Receive the boundaries *****/
    info=pvm_prekv(ids[tri],F4, f.f4[small_dim-1],dim,
                    PVM_FLOAT,&atid,&atag,&alen);
    info=pvm_prekv(ids[t1e],F2, f.f2[0],dim,
                    PVM_FLOAT,&atid,&atag,&alen);
}

/*****
/* INITIALIZATION PROCEDURE */

```

```

/*****
void INIT()
{
    int i;

/***** Memory allocation: *****/
    _f.f1=(float *) malloc((dim*small_dim)*sizeof(float ));
    _f.f2=(float *) malloc((dim*small_dim)*sizeof(float ));
    _f.f3=(float *) malloc((dim*small_dim)*sizeof(float ));
    _f.f4=(float *) malloc((dim*small_dim)*sizeof(float ));

    f.f1 =(float **)malloc(small_dim*sizeof(float *));
    f.f2 =(float **)malloc(small_dim*sizeof(float *));
    f.f3 =(float **)malloc(small_dim*sizeof(float *));
    f.f4 =(float **)malloc(small_dim*sizeof(float *));

    _integral=(float *) malloc((dim*small_dim)*sizeof(float ));
    _delay   =(int   *) malloc((dim*small_dim)*sizeof(int   ));
    _coeff.c1=(float *) malloc((dim*small_dim)*sizeof(float ));
    _coeff.c2=(float *) malloc((dim*small_dim)*sizeof(float ));

    integral =(float **) malloc(small_dim*sizeof(float *));
    delay     =(int   **) malloc(small_dim*sizeof(int   **));
    coeff.c1  =(float **) malloc(small_dim*sizeof(float **));
    coeff.c2  =(float **) malloc(small_dim*sizeof(float **));

/***** Built up arrays: *****/
    for(i=0;i<small_dim;++i){
        integral[i] = _integral+i*dim;
        delay   [i] = _delay  +i*dim;
        coeff.c1[i] = _coeff.c1+i*dim;
        coeff.c2[i] = _coeff.c2+i*dim;
        f.f1    [i] = _f.f1   +i*dim;
        f.f2    [i] = _f.f2   +i*dim;
        f.f3    [i] = _f.f3   +i*dim;
        f.f4    [i] = _f.f4   +i*dim;
    }

/**** Initialization *****/
    memset(_integral,0,dim*small_dim*sizeof(float));
    memset(_delay   ,0,dim*small_dim*sizeof(int  ));
    memset(_f.f1    ,0,dim*small_dim*sizeof(float));
    memset(_f.f2    ,0,dim*small_dim*sizeof(float));
    memset(_f.f3    ,0,dim*small_dim*sizeof(float));
    memset(_f.f4    ,0,dim*small_dim*sizeof(float));

/**** Build up the coefficients (including the border) *****/
/**** Every body *****/
    for(i=0;i<small_dim*dim;_coeff.c1[i++]=1.0);

```

```

for(i=0;i<small_dim*dim;_coeff.c2[i++]=0.5);

for(i=0;i<small_dim;i++){
    coeff.c1[i][0] =coeff.c1[i][0] *C0;
    coeff.c1[i][dim-1] =coeff.c1[i][dim-1]*C0;
}
for(i=1;i<small_dim-1;i++){
    coeff.c1[i][1] =coeff.c1[i][1] *C1;
    coeff.c1[i][dim-2] =coeff.c1[i][dim-2]*C1;
}
for(i=2;i<small_dim-2;i++){
    coeff.c1[i][2] =coeff.c1[i][2] *C2;
    coeff.c1[i][dim-3] =coeff.c1[i][dim-3]*C2;
}

/***** first task *****/
if(tno==0){
    for(i=0;i<dim;i++) coeff.c1[0][i] =coeff.c1[0][i] *C0;
    for(i=1;i<dim-1;i++)coeff.c1[1][i] =coeff.c1[1][i] *C1;
    for(i=2;i<dim-2;i++)coeff.c1[2][i] =coeff.c1[2][i] *C2;
}

/***** last task *****/
if(tno==nprocs-1){
    for(i=0;i<dim;i++) coeff.c1[small_dim-1][i]
        =coeff.c1[small_dim-1][i]*C0;
    for(i=1;i<dim-1;i++)coeff.c1[small_dim-2][i]
        =coeff.c1[small_dim-2][i]*C1;
    for(i=2;i<dim-2;i++)coeff.c1[small_dim-3][i]
        =coeff.c1[small_dim-3][i]*C2;
}
}

/*****
/* TIMING PROCEDURE */
/*****
int second()
{
    time_t timval;
    struct tm *tmptr;

    time(&timval);
    tmptr = localtime(&timval);

    return      tmptr->tm_sec
        +      60*tmptr->tm_min
        +      60*60*tmptr->tm_hour
        +      24*60*60*tmptr->tm_yday;
}

```

A *parflow* example

C.1 introduction

The purpose of this appendix is to present *Parflow* and to show how it is working by mean of a simple calibration problem, namely so-called four-corner problem. After the delivery of the software *Parflow 1.0* designed by the Group for Parallel Computing, it has been proposed to the Swiss Telecom PTT to develop an advanced version in order to improve the performance of the software and include more features.

Parflow 3.1 was delivered and installed on a Sun Sparc Station of the TELECOM PTT in Bern on 6th December 1996. *Parflow 3.1* includes the state-of-the-art in wave propagation using a lattice Boltzmann method: special boundary conditions, “fast TLM” (see section 4.5) simulation mode, Kirschhoff’s method (see section 4.4) and a real time visualization tool. The parameters are well defined and easy accessible in order to facilitate the enclosing of the software in the interface already used by Telecom PTT. A technical report [45] summarizes the key ingredients of the method but also serves as a user’s guide more detailed than this one.

C.2 The parameters of the simulation

The four-corner problem is based on an urban layout corresponding to a simple square. It is composed of four square-shaped buildings each of them with four walls. A compact `ascii` file contains the coordinates of the buildings’ corners given in a natural, or topographic unit (here 0.1 m). The format of this file is a

standard adopted by the Swiss Telecom PTT in which each building is defined by $N + 1$ lines where N is the number of corners of the building. The first line is a header that may contain various indentifiers such as the indices of the building, its height, its characteristic reflection coefficient, etc.; in our implementation, we have 2 fields: an ordering number (with a minus sign) and the number of corners (*i.e.* N). The following N lines give the couple of (x, y) coordinates of each building's corner in a coherent order, namely as encountered by following a path going continuously around it. There are 2 additional lines at the beginning of the file that gives the coordinates of the lower left and upper right corner of the bounding box embedding the whole layout. For our example we have:

```

0      0      # Bounding box (ll) natural unit = 0.1 m
2000  2000  # Bounding box (ur)
-1     4     # building No 1
1000  1233
1000  2000
2000  2000
2000  1233
-2     4     # building No 2
860   1265
860   2000
0     2000
0     1265
-3     4     # building No 3
0     1000
860   1000
860   0
0     0
-4     4     # building No 4
1000  1000
2000  1000
2000  0
1000  0

```

Basically, there is no need for us to have the buildings so properly defined: a scanned picture of an accurate map would be sufficient. But those kind of vectors' file are available because other methods need them. The measurement path is defined as a broken line given by the successive coordinates of its breaking points:

```

965   870
965   1070
1365  1070

```

There is one parameter file containing all the parameters. The file may content empty lines or comments if the first character is $\#$. The parameter line has two

letters labeling the parameter followed by the value and possibly followed by some comments. Because of the presence of these two letters the parameters may be given in any order. The parameter file can be edited and changed by the user before each run. Some parameters are predefined: the discrete period of the source $T = 8$, the associated amplitude factor $\gamma = 0.304$ and the set of three absorbing factors applied to the boundary of the domain: $(\mu_0, \mu_1, \mu_2) = (0.0, 0.436, 0.8)$. These parameters are fixed once for all because no improvement is expected out of tuning them. The remaining parameters used for the simulation presented on figure 4.9 are given below; note that [natural units] means the natural units (integers) used in the vector files defining the buildings and the measurement line (here 0.1 m). The list is given here without much comments and details are to be found in [45].

```

lo 40      # loop number
di 600     # Dimension i of the working array (verticale)
dj 600     # Dimension j of the working array (horizontale)
mi 0.6     # Automatic source postioning factor ]0,0.5]
mj 0.5     # Automatic source postioning factor ]0,0.5]
re .7      # Overall coefficient (reflexion coefficient)
tr .0      # PSI coefficient      (wall: 0 or 1/2)
dr 3.0     # Delta_r [natural unit]
la 1.5     # real wave length [natural unit]
th 45.0    # rotation angle [deg]
sx 985     # source position x [natural unit]
sy 700     # source position y [natural unit]
or c       # display result option
ld 100.0   # lowest db level for colormap
hd 50.0    # highest db level for colormap
df 5       # diffusion steps for averaging

```

Before the program was run it was compiled with some options to customize the executable. This customization consists of defining named identifiers with `#define`. The possible identifier are:

- **SHOW** This launches a visualization (xv) tool to display the output image in PGM format with a pre-defined system call.
- **KIR** This enables the experimental Kirschhoff's mode as described in section 4.4.
- **X11** This launches a graphical X-window displaying in real time the propagation intensities with a predefined color map. Simplified routines from the X1k library are called. X1k was developed separately by the author to provide an efficient visualization tool adapted to general purpose parallel machines.

- **ASCII** This converts the resulting image (PGM character format) in an ASCII form. The converted image can still be read by any viewer because it respects the ASCII ppm format. After header and comments one value is given by line, in a row by row order.
- **SIZE** This launches the system command `:ps -e -o vsz` to ask for the effective size in Kbytes occupied by the program because this is a critical value. It is written for SunOs 5.5.1 and the call appear in `lib/main.h`. Not very portable!

C.3 The results of the simulation

The simulation results may be extracted as an image representing the predicted intensity throughout the layout or as a file giving the intensity along the pre-defined measurement axis. Two different image formats are available: PGM or ASCII. For demonstration purpose or during an interactive session, the user may choose to visualize the progress of the simulation in “real-time”.

Bibliography

- [1] Pierre Thullier. *L'irrationnel dans la pensée scientifique*. Belin, 1997.
- [2] Serge Galam. Social paradoxes of majority rule voting and renormalization group. *J. of Stat. Phys.*, 61(3/4):943–951, 1990.
- [3] Serge Galam. Fragmentation versus stability in bimodal coalitions. *Physica A*, 230:174–188, 1996.
- [4] J. Hardy, Y. Pomeau, and O. de Pazzis. Molecular dynamics of classical lattice gas: Transport properties and time correlation functions. *Phys. rev. A*, 13:1949–1960, 1976.
- [5] U. Frisch, B. Hasslacher, and Y. Pomeau. Lattice-gas automata for the navier-stockes equation. *Phys. Rev. Lett.*, 56:1505, 1986.
- [6] Bastien Chopard and Michel Droz. *Cellular Automata Modelling of Physical Systems*. Cambridge University Press, 1998. In Press.
- [7] B. Chopard, P. Luthi, and M. Droz. Reaction-diffusion cellular automata model for the formation of liesegang patterns. *Phys. Rev. Lett.*, 72(9):1384–1387, February 1994.
- [8] Pascal O. Luthi. Cellular automata approach to liesegang's band. Master's thesis, Swiss Federal Institute of Technology, February 1993.
- [9] B. Chopard, P. O. Luthi, and M. Droz. Microscopic approach to the formation of liesegang patterns. *J. of Stat. Phys.*, 76(1/2):661–677, jul 1994.
- [10] R. E. Liesegang. *Naturw. Wochenschr.*, 11:353, 1896.
- [11] Pascal O. Luthi, Jeremy J. Ramsden, and Bastien Chopard. Role of diffusion in irreversible deposition. *Phys. Rev. E*, 55(3):3111–3115, march 1997.

- [12] Pascal O. Luthi, Annette Preis, Bastien Chopard, and Jeremy J. Ramsden. A cellular automaton model for neurogenesis in *drosophila*. To appear soon in *Physica D*, 1998.
- [13] B. Chopard, P.O. Luthi, and P.-A. Queloz. Cellular automata model of car traffic in two-dimensional street networks. *J. Phys. A*, 29:2325–2336, 1996.
- [14] C. Ngo and H. Ngo. *Physique Statistique à l'équilibre et hors d'équilibre*. Masson, 1995.
- [15] G.G. McNamara and G. Zanetti. Use of the boltzmann equation to simulate lattice-gas automata. *Phys. Rev. Lett.*, 61:2332–2335, 1988.
- [16] F. Higuera, J. Jimenez, and S. Succi. *Europhys. Lett.*, 9:663, 1989.
- [17] Christian Grossetête. *Mécanique des Fluides*. Ellipses Edition Marketing, 1991.
- [18] J. D. Jackson. *Classical Electrodynamics*. John Wiley & Sons, Inc, 1975.
- [19] Y. H. Quian, D. d'Humièrre, and P. Lallemand. Lattice bgk models for navier-stokes equations using a lattice-gas boltzmann method. *Europhys. Lett.*, 17(6):470–484, 1992.
- [20] P. Bhatnager, E. P. Gross, and M.K. Krook. *Phys. Rev.*, 94:511, 1954.
- [21] Jaroslaw Piasecki. *Echelles de temps en theorie cinétique*. Presses polytechnique et univerristaire romande, 1997.
- [22] Senior Editor Gary D. Doolen, editor. *Lattice Gas Methods for Parital Differential Equations*, Los Alamos, New Mexico, 1990. Santa Fe Institute, Addison Wesley.
- [23] S. Hou, J. Sterling, S. Chen, and G.D. Doolen. A lattice boltzmann subgrid model for high reynolds number flows. *Fields Institute Communications*, 6:151–166, 1996.
- [24] Wolfgang J. R. Hoeffler. The transmission-line matrix method. theory and applications. *IEEE Transaction on microwave theory and techniques*, MTT-33(10):882–893, oct 1985.
- [25] G. Kron. Equivalent circuit of the field equations of maxwell. *Procs IRE*, 32:289–299, May 1944.
- [26] H. J. Hrgovčić. Discrete representation of the n -dimensional wave equation. *J. Phys. A*, 25:1329–1350, 1991.

- [27] P. Sebbah C. Vanneste and D. Sornette. A wave automaton for time-dependent wave propagation in random media. *Europhys. Lett.*, 17(8):715–720, feb 1992.
- [28] D. Sornette P. Sebbah and C. Vanneste. A “wave automaton” for propagation in the time domain: I. periodic systems. *J. Phys. I France*, 3:1259–1280, jun 1993.
- [29] D. Sornette P. Sebbah and C. Vanneste. A “wave automaton” for propagation in the time domain: II. random systems. *J. Phys. I France*, 3:1281–1302, jun 1993.
- [30] B. Chopard and M. Droz. Cellular automata model for the diffusion equation. *J. of Stat. Phys.*, 64(3/4):859–892, August 1991.
- [31] P. Kuonen, F. Guidec, and P. Calegari. Object-oriented parallel software for radio wave propagation simulation in urban environment. Euro-Par’97, Aug 1997.
- [32] Al Geist, Adam Beguelin, Jack Dongarra, Weicheng Jiang, Robert Manchek, and Vaidy Sunderam. *PVM: Parallel Virtual Machine. A Users’ Guide and Tutorial for Networked Parallel Computing*. The MIT Press, Cambridge, Massachusetts, 1994.
- [33] Marc Martin and Bastien Chopard. Low cost parallelizing, a way to be efficient. In *Proceedings of the Third International Meeting on Vector and Parallel Processing*, Jun 1998. Submitted.
- [34] Pascal O. Luthi, Bastien Chopard, and Jean-Frédéric Wagen. Radio wave propagation simulator on scalable parallel computers. In Nicolas Droux, editor, *SIPAR Workshop ’95*. Biel School of Engineering, 1995.
- [35] H. L. Bertoni, W. Honcharenko, L. R. Maciel, and H. Xia. UHF propagation prediction for wireless personal communications. In *IEEE Proceedings, Vol 82, No. 9*, pages 1333–1359, 1994.
- [36] T. Kürner, D. J. Cichon, and W. Wiesbeck. Concepts and results for 3d digital terrain-based wave propagation models: an overview. *IEEE Jour. on Selected Areas in Communications*, JSAC-11(7):1002–1012, Sept. 1993.
- [37] K. Rizk, J.-F. Wagen, and F. Gardiol. Ray tracing based path loss prediction in two microcellular environments. In *Proceedings IEEE PIMRC’94*, pages 210–214, The Hague, Netherlands, sep 1994.
- [38] B. Chopard, Y. Baggi, and P.O. Luthi. Wave propagation and optimal antenna layout using a genetic algorithm. *Speedup Journal*, 1997. Proceedings of the Conference TelePar 97.

- [39] Y. Baggi. Procédé de planification en cellules urbaines. Technical report, CUI-Université de Genève, 1996. Master Dissertation.
- [40] STORMS. (software tools for the optimization of resources in mobile systems). is an ACTS project funded by the European Community and the Swiss government.
- [41] P.J. Cullen, S. Josselin, P. Kuonen, M. Pizarroso, and D. Wagner. Coverage and interference prediction and radio planning optimisation. volume 2, pages 557–562. ACTS Mobile Communication SUMMIT'97, October 1997.
- [42] P. Calegari, F. Guidec, and P. Kuonen. Urban radio network planning for mobile phones. *EPFL-Supercomputing Review*, 9, Nov 1997.
- [43] B. Chopard, P. Luthi, and J. F. Wagen. A lattice boltzmann method for wave propagation in urban microcells. *IEE Proc.-Microw. Antennas Propag.*, 144(4):251–255, August 1997.
- [44] H. Leutwyler. *Elektodynamik und Optik*. Institut für Theoretische Physik, Universität Bern, 1986.
- [45] Pascal Olivier Luthi. Parflow ii project - technical report. Technical report, CUI - University of Geneva, [luthi/chopard] @ cui.unige.ch, 1997.
- [46] J.-F. Wagen. Paper for in-berns measurements. In *Proceedings PIRMC'94*, The Hague, Netherlands, September 1994.
- [47] J. Li, J.-F. Wagen, and E. Lachat. A preliminary comparison of two propagation prediction methods for 2-d urban microcells: Ray tracing and tlm. In *Proceedings 46th IEEE Vehicular Technology Conference*, Chicago, Illinois, pp 859-863, April 1996.
- [48] M. Marder and Xiangming Liu. Instability in lattice fracture. *Phys. Rev. Let.*, 71(15):2417–2420, oct 1993.
- [49] S. J. Zhou, D. M. Beazley, P. S. Lomdahl, and B. L. Holian. Large-scale molecular dynamics simulations of three-dimensional ductile failure. *Phys. Rev. Let.*, 78(3):479–482, jan 1997.
- [50] Farid Abraham. Cracking a tough nut. IBM Research Division, Almaden Research Center: <http://www.tc.cornell.edu/farid/fracture/100million>, 1996.
- [51] Bastien Chopard. A cellular automata model of large-scale moving objects. *J. Phys. A: Math. Gen.*, 23:1671–1687, 1990.
- [52] M. Marder and J. Fineberg. How things breaks. *Physics Today*, pages 24–29, sep 1996.

- [53] Ping Sheng. *Introduction to Wave Scattering, Localization, and Mesoscopic Phenomena*. Academic Press, 1995.
- [54] Bert van Velzen. *Lattice Lorentz gases*. PhD thesis, Instituut voor theoretische fysica, Rijksuniversiteit Utrecht, 1990.

Index

A		D	
amplitude	65	data motion	47
anisotropy	37, 66	data parallel	51
array features	49	dB	65
array of data	48	deformation energy	88
array of pointers	48	diffusive contributions	23
		diffusive propagation	42
B		diffusive transport	104
bandwidth	51	disordered	103
barrier	57	dispersion relation	33
benchmark	45	dissipation	26, 28, 96
black and white	86	distributed memory	50
blocking MP	54	Drosophila	5
Boltzmann equation	8	ductile	99
brittle	99	dynamical memory allocation ...	44
C		E	
C	49	effective medium	104
causality context	46	effective reflection coefficient ...	68
causality disc	76	Einstein	43
Cellular Automata	4	EPFL	64
cellular phones	63	F	
CM200	52	F77	47
CMF	52	F90	45, 47
coherent backscattering effect ...	105	fast deviations	81
collective behavior	8	floating-point operations	46
collision term	15	Flop	46
conservation laws	20	focusing effect	29
Conway	4	Fortran 77	49
coordination number	14	Fortran 90	49
crack dynamics	99	four-corner problem	81
critical stress	99	fracture	86, 98
cshift	47		

frequency	66, 73
G	
generalized charge density	14
generalized current density	14
genetic algorithm	64
gravitational potential	98
H	
Hamiltonian formalism	12
heterogeneous media	63
HPF	53
huyghens' principle	74
I	
IBM SP2	53
index of refraction	24
K	
Kepler	2
Kirchoff integral	74
knife-wedge problem	79
L	
large scale objects	85
lattice BGK	16
lattice orientation	83
Liesegang patterns	4
load balancing	52
local calculation	43
local equilibrium	9
locality	38
localization length	105
Lorentz gas	105
M	
Manhattan distance	71
mass of rest particle	32
master-slave	56
Maxwell equations	12
Maxwell stress tensor	13
mean delay time	71
memmove	47
memory hierarchies	50
message passing	53

MIMD	43
mode I loading	102
molecular dynamics	86
momentum	89
MPI	54
MPMD	54
MPP	43
multi-scale analysis	19

N

nearest neighbors	43
Newton	2
non-blocking MP	54
normalization factor	66

O

optimization problem	64
----------------------------	----

P

parabolic mirror	29
parallelization	50
Parflow	64
perfect absorbing boundary	69
perfect reflection	29
performance model	51
phase	65
point-to-point communication	54
potential energy	88
Poynting vector	13
pseudo 3D	70
PVM	54

Q

quadratic form	31
----------------------	----

R

radio wave propagation	64
ray tracing	64
ray-tracing method	78
reflection	95
reflectors	29
regular data structures	43
relaxation time	9
renormalization method	70

ring-shaped decomposition ... 75, 76

S

scaling law 51
 scattering matrix 40
 sector-shaped decomposition 75
 secular divergence 19
 shadow boundary 80
 shared memory 50
 shared memory model 57
 sheet of atoms 90
 SIMD 43
 simulation 2, 78
 sinusoidal source node 66
 site of attenuation 68
 site of reflection 67
 social physics 4
 solid body motion 93
 space inversion 31
 staggered invariants 34
 static stress 102
 statistical mechanics 8
 STORM 64
 stress tensor 20, 21
 string 86
 symmetrized matrix 35
 symmetry of W 40
 synchronization 54

T

task 57
 Taylor expansion in τ 18
 temperature 93
 thermodynamics 8
 thread 57
 time inversion 31
 time reversal invariance 25
 TLM 33
 total energy 92
 transmission line matrix 30

U

unitarity of W 39
 urban environment 63

V

virtual particle 91
 volume force 97
 von Neumann 4

W

wave equation 11
 wave length 66
 wave localization 103
 weak-localization effect 105
 Wolfram 4