Amélioration de la complexité en moyenne de l'algorithme de Lee et Schachter par division et fusion bi-directionnelles

Christophe LEMAIRE[1,2]
Jean-Michel MORAEU[2]

[1] SETRA, *46 avenue Aristide Briand, BP 100*
F 92 223 Bagneux Cedex.
[2] EMSE, *158 Cours Fauriel,*
F 42023 Saint-Étienne

RÉSUMÉ. *Nous avons cherché à améliorer les performances en moyenne de l'algorithme optimal (pour le pire des cas) de Lee et Schachter [LS 80] qui calcule la triangulation de Delaunay d'un ensemble de N points du plan avec une méthode du type "Diviser et fusionner". Pour cela, nous utilisons le découpage induit par un $2d$-arbre, qui peut être construit à partir d'un tri selon deux directions. Si la distribution des sites est quasi-uniforme dans un carré unité, nous montrons alors que la phase de triangulation (après la création du $2d$-arbre) est linéaire en moyenne alors que dans l'algorithme de Lee et Schachter, elle est en $\Theta(N \log N)$, même en moyenne [OIM 84].*
En pratique, le temps de triangulation est divisé par 2 ou 3. Sur une Silicon Graphics Indy (200 MHz), on triangule 200 000 points en 7 secondes après une phase de tri de 3 secondes. Cette méthode est plus rapide que les méthodes par balayage pour le même problème.

ABSTRACT. *We have tried to improve the average running time of the (worst case) optimal algorithm by Lee and Schachter [LS 80] that computes the Delaunay triangulation of N points in the plane, using a "Divide-and-Conquer" technique. To that effect, we use the division induced by a $2d$-tree that may be constructed from a two-directional sort. If the distribution of sites is quasi-uniform in a unit square, we show that the triangulation phase (after the construction of the $2d$-tree) takes linear average time, whereas it takes $\Theta(N \log N)$ time in Lee and Schachter's case, even in the average ([OIM 84]).*
In practice, the triangulation time is divided by 2 or 3. On a Silicon Graphics Indy (200 MHz), 200,000 sites are triangulated in 7 seconds, after a sorting phase of 3 seconds. This method runs faster than sweep-line techniques for the same problem.

MOTS-CLÉS : *triangulation de Delaunay, diagramme de Voronoi, tri, $2d$-arbre, complexité dans le pire des cas, complexité en moyenne.*

KEYWORDS: *Delaunay triangulation, Voronoi diagram, sort, $2d$-tree, worst-case running time, average running time.*

1 Introduction

Soient $\mathbb{E}^2$ le plan muni de la distance euclidienne d, et $S = \{M_1, M_2, \dots M_N\}$ un ensemble de N points de $\mathbb{E}^2$, appelés *sites*. On associe à chaque site M_i la région $V(M_i)$ de $\mathbb{E}^2$ des points du plan plus proches de M_i que de n'importe quel autre site :

$$V(M_i) = \{P \in \mathbb{E}^2 \mid d(P, M_i) \leq d(P, M_j), \forall j \in [1, n] - \{i\}\},$$

Pour tout $i \in [1, n]$, $V(M_i)$, qui est appelée la *région de Voronoi de centre* M_i, est l'intersection d'un nombre fini de demi-plans limités par les médiatrices des segments $[M_i M_j]$, $j \in [1, n] - \{i\}$. C'est donc un polygone convexe, éventuellement non borné, appelé *polygone de Voronoi de centre* M_i.

L'union des $V(M_i)$ constitue une subdivision de $\mathbb{E}^2$, appelée le *diagramme de Voronoi* de S, noté $DV(S)$. L'intersection de deux polygones de Voronoi quelconques de centre respectif M_i et M_j est soit vide (la médiatrice du segment $M_i M_j$ n'apparaît pas dans le diagramme), soit réduite à un point (M_i et M_j sont situé sur le même cercle d'intérieur vide qu'au moins deux autres sites de S), soit enfin égale à une section non dégénérée de la médiatrice de $M_i M_j$. Dans ce dernier cas, on dit que les sites M_i et M_j sont *voisins* au sens de Voronoi.

Le *dual* (faible) de $DV(S)$ est un graphe *planaire* $T = (S, A, F)$, appelé *diagramme de Delaunay*, qui forme une partition de l'enveloppe convexe de S, et dont l'ensemble des arêtes est

$$A = \{(M_i, M_j) \in S^2, i \neq j \mid M_i, M_j \text{voisins au sens de Voronoi}\},$$

et dont les faces F sont les polygones constitués par ces arêtes.

Tous les sommets définissant la frontière d'un même polygone de Delaunay sont cocycliques. Le diagramme de Delaunay devient une *triangulation* et est unique si et seulement si toutes ses faces sont des triangles, *i.e.* S ne contient aucun quadruplet de sites situés sur un même cercle d'intérieur vide. Sinon, on peut toujours obtenir "une" triangulation de Delaunay de S en rajoutant arbitrairement les cordes nécessaires dans les faces non-simpliciales de F. Par abus de langage, on parlera toujours de *la* triangulation de Delaunay de l'ensemble S, que l'on notera $TD(S)$.

Les diagrammes de Voronoi et les triangulations de Delaunay sont des structures très puissantes, souvent utilisées pour les problèmes de proximité, la planification de mouvements, le calcul par éléments finis, la représentation de terrains naturels ... Pour un survol complet de ces deux structures et de leurs propriétés, se reporter, par exemple, à l'article de F. Aurenhammer [AUR 91].

Une des propriétés remarquables de la triangulation de Delaunay est le *critère du cercle vide*: le cercle circonscrit à un triangle de $TD(S)$ ne contient aucun site de S en son intérieur strict. Ceci constitue un critère de construction efficace, que Lee et Schachter ont exploité en 1980 pour obtenir le premier algorithme optimal de construction de la triangulation de Delaunay de N sites du plan en temps $O(N \log N)$.

En 1984, A. Maus ([MAU 84]) publia la première méthode linéaire en moyenne de construction de la triangulation de Delaunay d'un ensemble de sites du plan, à l'aide d'une grille pour accélérer la localisation du voisin le plus proche vérifiant le critère du cercle vide pour toute arête de Delaunay donnée.

En 1987, R. Dwyer publia une méthode de complexité $O(N \log \log N)$ en moyenne ([DWY 87]) pour le même problème dans le cas d'une distribution uniforme, à l'aide d'une grille régulière. Dans leur article de 1988, [KK 88], J. Katajainen et M. Koppinen introduisirent une variante de l'algorithme standard de Lee et Schachter, s'appuyant encore sur une grille régulière. Leur analyse de complexité en moyenne introduit et exploite la notion de *site inachevé* dans un domaine rectangulaire, qui est reprise dans cet article. Enfin, R. Dwyer publia la première méthode k-dimensionnelle de complexité moyenne linéaire en 1991 ([DWY 91]).

Par opposition, la méthode décrite dans le présent article ne découpe pas le plan à l'aide d'une grille, mais selon une arborescence équilibrée. L'article est construit de la manière suivante : la section 2 rappelle les points saillants de l'algorithme classique dû à Lee et Schachter pour la construction de la triangulation d'un ensemble de sites du plan selon le paradigme *Division-fusion*. La section 3 présente la technique de division et fusion dans deux directions et l'algorithme qui en résulte, dont la section 4 évalue la complexité asymptotique dans le pire des cas et en moyenne pour une distribution quasi-uniforme des sites dans un carré unité. La section 5 présente quelques résultats expérimentaux obtenus pour l'algorithme présenté dans cet article, qui se termine par quelques remarques de conclusion, section 6.

2 Division et fusion uni-directionnelles

Cette section présente le principe sous-tendant l'algorithme original de Lee et Schachter ([LS 80]) pour la construction de la triangulation de N sites dans le plan. Pour simplifier, on fait l'hypothèse que la triangulation de Delaunay est unique (*cf.* Introduction).

2.1 *Division de l'ensemble des sites*

On choisit une direction du plan privilégiée, par exemple verticale. L'ensemble des sites est divisé récursivement en deux parties séparables par une droite verticale [1], et de même taille (à une unité près), jusqu'à ce que les sous-ensembles ne contiennent qu'un seul site (Figure 1).

Ce découpage équilibré du plan nécessite la réorganisation, à chaque niveau de récursion, de l'ensemble courant de part et d'autre de la X-médiane, ce qui peut être fait en temps optimal linéaire par l'algorithme de sélection de Blum *et al.* ([BFPRT 73]). Mais il est bien connu qu'à complexité asymptotique globale identique, il est bien plus efficace de pré-trier les sites de S selon le xy-ordre : selon les x croissants, et en cas d'égalité, selon les y croissants. Une fois ce tri opéré, l'accès à la médiane peut être effectué en temps constant.

Ce choix de la direction de division (et fusion) est arbitraire, et l'on serait tout autant fondé à choisir la direction horizontale. En fait, tout ce qui est dit dans cette section et la suivante est valable pour l'une ou l'autre direction.

1. En fait, cette droite peut parfois être légérement oblique pour séparer les points ayant même abscisse.

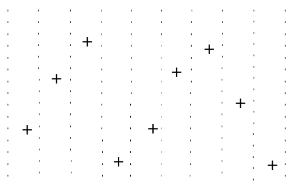

FIG. 1 – *Division récursive en bandes élémentaires.*

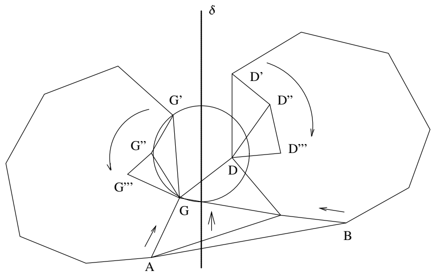

FIG. 2 – *Fusion de deux triangulations linéairement séparables.*

2.2 *Fusion des sous-ensembles élémentaires*

Les bandes élémentaires adjacentes sont alors fusionnées par paires de telle sorte que la triangulation des points contenus dans chaque bande résultante soit de Delaunay. Ce processus est itéré jusqu'à obtention de la triangulation de Delaunay de l'ensemble S total.

Pour fusionner deux triangulations $TD(G)$ et $TD(D)$ linéairement séparables par une droite δ verticale, on part des sites de G et de D qui sont les plus proches de cette droite, et l'on progresse dans le sens horaire (*resp.* direct) sur la frontière de l'enveloppe convexe de G (*resp.* D), jusqu'à trouver les extrémités (A et B sur la Figure 2) de l'arête la plus basse de l'enveloppe convexe de $G \cup D$ qui coupe δ. Ensuite, les nouvelles arêtes sont créées dans l'ordre selon lequel elles coupent δ jusqu'à la création de l'arête la plus haute de l'enveloppe convexe de $G \cup D$ coupant δ.

Les arêtes de $TD(G)$ et $TD(D)$ qui coupent les arêtes créées pendant la fusion, sont détruites. En fait, dès qu'une arête GD est créée, on réalise les opérations suivantes :

a *Nettoyage à gauche autour du site* G*, dans le sens direct :* Tant que G' est situé au-dessus de GD, on élimine l'arête GG' si D est inclus dans l'intérieur de $\bigcirc(G, G', G'')$, puis on passe à GG'', sinon on arrête.

b *Nettoyage à droite autour du site* D*, dans le sens horaire :* Tant que D' est situé au-dessus de GD, on élimine l'arête DD' si G est inclus dans l'intérieur de $\bigcirc(D, D', D'')$, puis on passe à DD'', sinon on arrête.

Dans le quadrilatère $GG'D'D$ résultant de ces nettoyages, on choisit la diagonale satisfaisant le critère du cercle vide ("au hasard" si $GG'D'D$ est inscriptible dans un cercle), et ce choix induit la création d'une nouvelle arête (GD' ou $G'D$) qui devient la nouvelle GD dans l'itération suivante, et ainsi de suite…

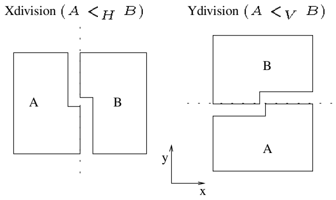

FIG. 3 – *X- et Y-divisions.*

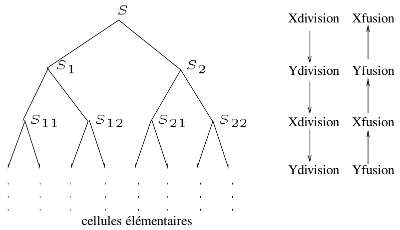

FIG. 4 – *Arborescence schématisant les divisions et les fusions.*

3 Division et fusion bi-directionnelles

Le découpage récursif des N sites selon une direction unique produit des bandes élémentaires très longues et très étroites. La fusion de ces bandes engendre un assez grand nombre de triangles qui n'ont pratiquement aucune chance d'appartenir à la triangulation finale, étant donné leur forme excessivement allongée (partie gauche de la Figure 17).

L'algorithme perd beaucoup de temps, au fur et à mesure des fusions entre bandes adjacentes de même direction, à détruire et reconstruire des triangles. Ohya, Iri et Murota ([OIM 84]) ont montré que la complexité de l'étape de fusion est, *même en moyenne*, en $\Omega(N \log N)$. Ceci est fondamentalement dû au principe de découpage par bandes uni-directionnelles.

Pour optimiser l'algorithme précédent, nous proposons de diviser le plan selon un autre processus, qui évite la création de bandes étroites : la division (et la fusion) selon deux directions orthogonales (partie droite de la Figure 17).

3.1 *X- et Y-divisions*

Notons M_x et M_y l'abscisse et l'ordonnée d'un point M, et définissons les relations $<_H$ et $<_V$ respectivement par :

$$\begin{array}{lcl} M <_H N & \Leftrightarrow & ((M_x < N_x) \vee (M_x = N_x \wedge M_y < N_y)), \quad \text{et} \\ M <_V N & \Leftrightarrow & ((M_y < N_y) \vee (M_y = N_y \wedge M_x > N_x)) \end{array}$$

Soient A et B deux ensembles de points de $\mathbb{E}^2$. On dira que $A <_H B$ (*resp.* $A <_V B$) si et seulement si tous les points de A sont $<_H$ (*resp.* $<_V$) à tous les points de B.

En se référant à la Figure 3, on dira que l'on X-divise (*resp.* Y-divise) un ensemble A de points du plan en A_1 et A_2 si et seulement si $A_1 <_H A_2$ (*resp.* $A_1 <_V A_2$), $A_1 \cup A_2 = A$ et si A_1 et A_2 ont le même nombre de points, à une unité près. On remarquera que A_1 ou A_2 peut être vide.

Les sites vont être divisés selon la dynamique d'un $2d$-*arbre* ([BEN 75]) *équilibré*. Un $2d$-arbre est obtenu en divisant alternativement le plan suivant les directions horizontales et verticales. Plus précisément, on commence par X-diviser l'ensemble S en S_1 et S_2. Ensuite, on Y-divise S_1 (*resp.* S_2) en S_{11} et S_{12} (*resp.* S_{21} et S_{22}). Ensuite,

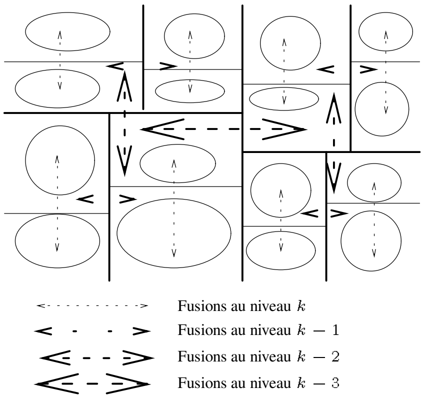

FIG. 5 – *Processus de fusion.*

les parties S_{11}, S_{12}, S_{21} et S_{22} sont X-divisées. On itère ce processus en alternant les X-divisions et les Y-divisions jusqu'à l'obtention de cellules élémentaires contenant un seul site (Figure 4).

L'équilibrage de l'arbre de division est garanti par le "partitionnement" du sous-ensemble relatif au nœud courant autour de la X- ou Y-médiane. Comme pour l'algorithme original, il existe plusieurs façons de procéder pour effectuer ces partitions : on peut utiliser l'algorithme linéaire déterministe de sélection de la médiane de Blum *et al.* (*cf.* [BFPRT 73]), ou bien une méthode dérivée de *Quicksort*, randomisée ([CLR 94]) ou non ([HOU 87] [2]), ou encore tout simplement effectuer un tri de prétraitement dans les deux directions. Quelle que soit la méthode choisie, on supposera sa complexité théorique globale en $O(N \log N)$.

3.2 X*- et* Y*-fusions*

Les cellules élémentaires sont ensuite fusionnées par paires dans l'ordre inverse de leur création. Les sous-ensembles obtenus par X-division (*resp.* Y-division) sont fusionnés par X-fusion (*resp.* Y-fusion), (cf. Figure 5).

Comme nous l'avons déjà signalé, cela est possible car la procédure de fusion de Lee et Schachter est très générale et fonctionne de la même façon, quelle que soit la direction de la droite de séparation entre les deux triangulations à fusionner, à condition de choisir des structures de données bien adaptées.

Structure de données pour la division-fusion bi-directionnelle

Les procédures de fusion (et de division) présentent un assez grand degré de symétrie, nécessitant des structures de données flexibles et symétriques. La structure de "carte planaire" ([COR 75]), proche de celle d'arête ailée ([BAU 75]), aussi compacte

2. C'est la méthode que nous avons choisie pour son efficacité (en particulier dans le cas de distributions quasi-uniformes), sa simplicité, et qui correspond aux temps donnés dans la section 5.

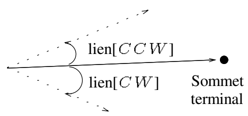

FIG. 6 – *Un* brin*, structure sous-tendant la "carte planaire".*

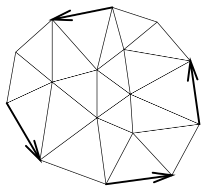

FIG. 7 – *Les quatre brins extrêmes d'une triangulation.*

et légérement plus souple, a été choisie pour représenter les graphes : un sommet est représenté par un couple de coordonnées; un "brin" (de la carte planaire) comprend un pointeur sur son sommet terminal, et (un tableau de) deux pointeurs sur les brins suivant et précédent (selon l'ordre polaire) autour de son propre sommet origine (implicite) (Figure 6). Les brins sont stockés dans un tableau, de sorte qu'un brin et son inverse aient des positions symétriques par rapport au milieu du tableau, ce qui dispense de tout lien explicite de dualité.

Pour fusionner deux triangulations linéairement séparables, on a besoin du site de chaque sous-triangulation le plus proche de la droite de séparation, que celle-ci soit verticale ou horizontale. Pour accéder à ces sommets en temps constant, on garde en mémoire (sous forme de matrice 2×2) les quatre brins extrêmes de chaque sous-triangulation, *i.e.* l'arc orienté d'enveloppe convexe issu de chaque sommet extremum pour les relations d'ordre $<_V$ et $<_H$ (*cf.* Figure 7).

3.3 *L'algorithme*

On suppose que les sites ont été rangés dans un tableau sans point double lors d'une Étape 0; g et d représentent les extrémités du tableau courant dans chaque fonction, et l'on pose $\mu = \lceil (g + d)/2 \rceil$. Enfin, $\varepsilon \in \{0, 1\}$ représente la "direction" courante (0 pour verticale, 1 pour horizontale).

Etape 1 : Tri sur place du tableau de sites selon deux directions

Fonction XY-Division (g, d, ε)
  Si $((d - g) > 0)$ **alors**
    XY-Partition(g, d, ε)
    XY-Division$(g, \mu - 1, 1 - \varepsilon)$
    XY-Division$(\mu, d, 1 - \varepsilon)$

Etape 2 : Triangulation

Fonction Delaunay(g, d)
  XY-Coupe$(g, d, 0)$
Fonction XY-Coupe(g, d, ε)
  Si $((d - g) > 0)$ **alors**
    XY-Coupe$(g, \mu - 1, 1 - \varepsilon)$
    XY-Coupe $(\mu, d, 1 - \varepsilon)$
    XY-Fusion(g, d, ε)

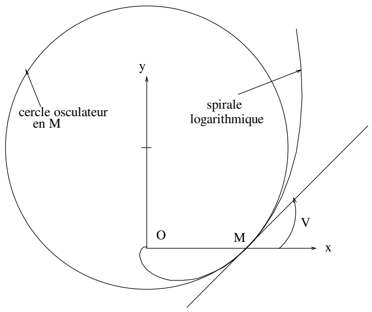

FIG. 8 – *Propriétés de la spirale logarithmique.*

4 Analyse de complexité

On s'intéresse ici à la complexité de la phase de triangulation, une fois le tri selon deux directions des N sites effectué. Ce tri peut être réalisé en $\Theta(N \log N)$, comme nous l'avons déjà fait remarquer.

4.1 *Complexité dans le pire des cas*

Nous allons montrer que la complexité dans le pire des cas est en $\Omega(N \log N)$. Pour cela, nous allons étudier la triangulation de Delaunay de N sites répartis sur une portion de spirale logarithmique.

La spirale logarithmique d'équation en coordonnées polaires $\rho = ae^{m\theta}$ a de nombreuses propriétés (Figure 8) :

1. La tangente en M fait un angle V constant avec la droite OM ($\tan V = \frac{1}{m}$).
2. Le cercle passant part M et centré au centre de courbure en M contient toute la section de la spirale situé *avant* (au sens de l'abscisse curviligne) M ; toute la section de la spirale *après* M est à l'extérieur de ce cercle.
3. Le cercle circonscrit à trois sites M_i, M_j et M_k $(i < j < k)$ situés sur la spirale contient toute la partie de la spirale *avant* M_k ; toute la partie de la spirale *après* M_k est à l'extérieur de ce cercle.
4. Il en résulte que construire la triangulation de Delaunay d'un ensemble de sites M_i situés sur une portion de spirale consiste à relier le site M_1 à tous les autres sites M_j $(j \neq 1)$, puis à relier chaque site M_i à son suivant M_{i+1} . Ainsi, si l'on ajoute un site sur la spirale avant M_1, il n'y a plus un seul triangle valide (Figure 9).

Considérons par exemple la spirale d'équation $\rho = e^{\theta}$.

$$\begin{cases} x(\theta) = e^{\theta}\cos\theta \\ y(\theta) = e^{\theta}\sin\theta \end{cases} \Rightarrow \begin{cases} x\prime(\theta) = \sqrt{2}e^{\theta}\cos(\theta + \frac{\pi}{4}) \\ y\prime(\theta) = \sqrt{2}e^{\theta}\sin(\theta + \frac{\pi}{4}) \end{cases}$$

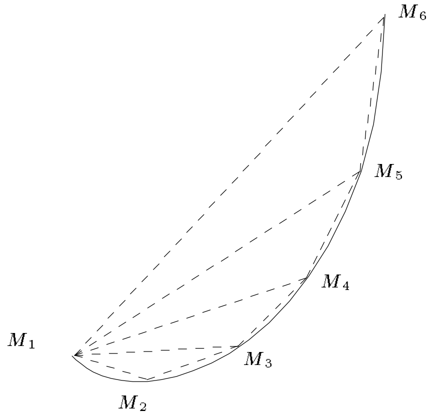

FIG. 9 – *Triangulation sur une portion de spirale.*

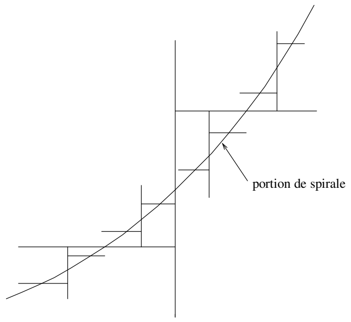

FIG. 10 – *Découpage induit par le $2d$-arbre.*

Les abscisses et les ordonnées sont strictement croissantes sur l'intervalle $]-\frac{\pi}{4}, \frac{\pi}{4}[$. Supposons que l'on répartisse N sites sur cette portion de spirale. Le $2d$-arbre va découper cet ensemble en cellules qui ont la propriété suivante : si l'on considère les sites d'une cellule, ils sont soit tous *avant* les sites d'une autre cellule (au sens des relations $<_V$ et $<_H$), soit tous *après* (Figure 10). Par conséquent, lors de la fusion de deux cellules, tous les triangles de l'une ou de l'autre seront détruits.

On a donc l'équation de récurrence suivante :

$$T(N) = 2T(\frac{N}{2}) + \alpha N \Rightarrow T(N) \in \Omega(N \log N)$$

L'algorithme a une complexité dans le pire des cas dans $\Omega(N \log N)$. $\diamond$

4.2 *Complexité en moyenne*

Pour calculer la complexité en moyenne, nous avons besoin de résultats établis par Guibas et Stolfi [GS 85], Dwyer [DWY 87], Katajainen et Koppinen [KK 88]. Ces résultats, ainsi que leur démonstration (parfois légèrement remaniée), sont rappelés dans les paragraphes 4.2.1, 4.2.2 et 4.2.3. Par contre, la partie 4.2.4 est originale et prouve que la triangulation se fait en $O(N)$ en moyenne.

On suppose que la distribution est quasi-uniforme sur un carré unité U. Soit f la densité de probabilité :

$$\begin{cases} \forall (x, y) \in U, & 0 < c_1 \leq f(x, y) \leq c_2 \\ \forall (x, y) \notin U, & f(x, y) = 0 \end{cases}$$

Par exemple, la probabilité qu'un point donné soit dans un domaine D vaut $\int_D f$.

4.2.1 *Sites inachevés*

Cette notion a été introduite par J. Katajainen et M. Koppinen dans [KK 88], et est primordiale dans la justification de la linéarité de la complexité en moyenne de l'algorithme décrit dans l'article présent. Nous en redonnons la définition :

Définition 4.1 *([KK 88]) Soient $T(S_1)$ et $T(S_2)$ des triangulations des ensembles S_1 et S_2. On note $T(S_1) <_\triangle T(S_2)$ si $S_1 \subseteq S_2$ et si $T(S_1)$ contient chaque arête de $T(S_2)$ dont les extrémités appartiennent à S_1.*

Cette définition crée un ordre partiel dans l'ensemble des triangulations. Remarquons que $T(S_1) <_\triangle T(S_2)$ n'implique pas que toutes les arêtes de $T(S_1)$ appartiennent à $T(S_2)$. Remarquons aussi que $TD(S_1) <_\triangle TD(S_2)$ est équivalent à $S_1 \subseteq S_2$ si S_1 a une triangulation de Delaunay unique.

Définition 4.2 *([KK 88]) Soient $T(S_1) <_\triangle T(S_2)$ deux triangulations et $s \in S_1$. On dit que s est achevé dans $T(S_1)$ relativement à $T(S_2)$ si l'ensemble des arêtes incidentes à s dans $T(S_1)$ et $T(S_2)$ coïncident, sinon on dit que s est inachevé.*

Proposition 4.1 *([KK 88]) Soient $T(S_1) <_\triangle T(S_2)$ deux triangulations. Un site $s \in S_1$ est achevé dans $T(S_1)$ relativement à $T(S_2)$ si et seulement si S_1 contient les extrémités de toutes les arêtes incidentes à s dans $T(S_2)$.*

Dém. : Soient (s, p_1),...(s, p_m) les arêtes adjacentes à s dans $T(S_2)$. Si s est achevé, alors les extrémités p_i appartiennent à S_1 par définition.

Réciproquement, supposons que les extrémités p_i appartiennent à S_1 mais que s ne soit pas achevé. Alors, (s, p_1),...(s, p_m) appartiennent à $T(S_1)$ parce que $T(S_1) <_\triangle T(S_2)$, mais $T(S_1)$ contient au moins une autre arête (s, r). Puisque le domaine délimité par $T(S_2)$ est convexe, il contient entièrement l'arête (s, r).

Par conséquent, $(s, r) \setminus \{s\}$ coupe, soit une des arêtes (s, p_i), soit un des triangles ouverts de $T(S_2)$ ayant s pour sommet. La première alternative est évidemment impossible. Un triangle de $T(S_2)$ ne contient aucun site de S_2 en son intérieur, et en particulier pas r. Donc, (s, r) coupe le côté opposé (p_i, p_j), ce qui est tout autant impossible car (p_i, p_j) appartient aussi à $T(S_1)$, qui est une triangulation. $\diamond$

Corollaire 4.1 *([KK 88]) Soient $T(S_1) <_\triangle T(S_2) <_\triangle T(S_3)$ trois triangulations. Un site $s \in S_1$ est achevé dans $T(S_1)$ relativement à $T(S_3)$ si et seulement s'il est achevé à la fois dans $T(S_1)$ relativement à $T(S_2)$ et aussi dans $T(S_2)$ relativement à $T(S_3)$.*

Dém. : La dernière condition implique la précédente à cause de la Définition 4.2.

Réciproquement, supposons que s soit achevé dans $T(S_1)$ relativement à $T(S_3)$. Si (s, q) est une arête dans $T(S_3)$, alors $q \in S_1$; comme $q \in S_2$, en utilisant la Proposition 4.1, on obtient que s est achevé dans $T(S_2)$ relativement à $T(S_3)$. Enfin, à cause de la Définition 4.2, s est aussi achevé dans $T(S_1)$ relativement à $T(S_2)$. $\diamond$

4.2.2 *Complexité d'une fusion en fonction du nombre de sites inachevés*

Théorème 4.1 *([GS 85]) Soient G et D deux ensembles de sites situés respectivement à gauche et à droite d'une droite δ. Lors de la fusion de $TD(G)$ et $TD(D)$ pour construire $TD(G \cup D)$:*

1. *Seules des arêtes reliant deux sites de G et des arêtes reliant deux sites de D sont détruites.*
2. *Seules des arêtes reliant un site de G à un site de D sont créées.*
3. *La complexité dans le pire des cas du processus de fusion est bornée par une fonction linéaire de la somme des trois composants suivants :*
   - *(a) Le nombre de sites examinés pour trouver les extrémités de l'arête la plus basse de l'enveloppe convexe de $(G \cup D)$ qui coupe δ.*
   - *(b) Le nombre d'arêtes détruites.*
   - *(c) Le nombre d'arêtes créées.*
4. *La complexité dans le pire des cas du processus de fusion est en $O(|G \cup D|)$.*

Dém. : cf. [GS 85]. ◇

Lemme 4.1 *([DWY 87]) Toute triangulation d'un ensemble de n sites dont k appartiennent à l'enveloppe convexe, possède $e = 3n - k - 3$ arêtes et $t = 2n - k - 2$ triangles. En particulier, au plus $3n - 6$ arêtes peuvent être créées sur un ensemble de n sites.*

Dém. : Soient t le nombre de triangles et e le nombre d'arêtes. Toutes les arêtes appartiennent à deux triangles sauf les k arêtes de l'enveloppe convexe, d'où $2e - k = 3t$. En utilisant la formule d'Euler $n - e + (t + 1) = 2$, on en déduit t et e. En faisant $k = 3$, on déduit le dernier résultat. ◇

Théorème 4.2 *([DWY 87]) La complexité dans le pire des cas du processus de fusion est bornée par une fonction linéaire du nombre de sites qui reçoivent de nouvelles arêtes.*

Dém. : Grâce au Théorème 4.1, il suffit de montrer que, si de nouvelles arêtes sont ancrées sur m sites, alors au plus $3m - 6$ arêtes sont créées, au plus $3m - 9$ arêtes sont détruites, et au plus m sites sont examinés pour construire l'arête la plus basse de l'enveloppe convexe de $G \cup D$ qui coupe δ.

Soient c et d le nombre d'arêtes créées et détruites. Puisque les arêtes créées ne se coupent pas, en raison du Corollaire 4.1, on a $c \leq 3m - 6$.

Supposons que G, D et $G \cup D$ soient composés respectivement de n_1, n_2 et n sites, dont k_1, k_2 et k appartiennent à l'enveloppe convexe. On peut tirer l'enchaînement logique suivant :

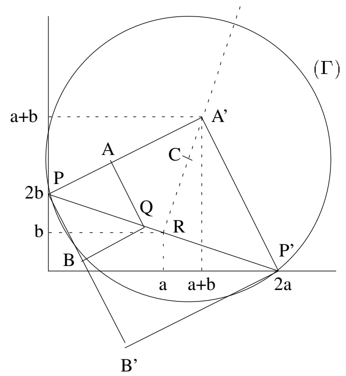

FIG. 11 – *Le cercle* (Γ) *contient* $\triangle PQA$ *ou* $\triangle PQB$.

$$\begin{array}{rl} \text{Lemme 4.1} \Rightarrow & (3n_1 - k_1 - 3) + (3n_2 - k_2 - 3) + (c - d) = (3n - k - 3) \\ n = n_1 + n_2 \Rightarrow & k - (k_1 + k_2) = 3 - c + d \\ \left.\begin{array}{r} k \leq k_1 + k_2 \\ 3 - c + d \leq 0 \end{array}\right\} \Rightarrow & d \leq c - 3 \\ c \leq 3m - 6 \Rightarrow & d \leq 3m - 9 \end{array}$$

Enfin, nous observons que chaque site examiné pour trouver l'arête la plus basse de l'enveloppe convexe de $G \cup D$ qui coupe δ, reçoit une nouvelle arête pendant la fusion, donc leur nombre est inférieur à m. ◇

Corollaire 4.2 *([DWY 87]) La complexité dans le pire des cas du processus de fusion est majorée par une fonction linéaire du nombre d'arêtes créées.*

Dém. : Évident. ◇

Corollaire 4.3 *La complexité dans le pire des cas de la fusion de* G *et* D *est majorée par* $7m$ *où* m *est le nombre de sites inachevés dans* G *et dans* D.

Dém. : Le nombre total d'arêtes créées est majoré par $3m$, le nombre total d'arêtes détruites est majoré par $3m$, le nombre total de sites examinés pour trouver l'arête la plus basse de l'enveloppe convexe de $L \cup R$ qui coupe δ, est majoré par m. ◇

4.2.3 *Nombre de sites inachevés dans un domaine rectangulaire*

Lemme 4.2 *([DWY 87]) Soient* P, Q *et* P' *trois points alignés dans cet ordre. Soient* A *et* B *les autres sommets du carré de diagonale* PQ. *Alors tout cercle* (Γ) *passant par* P *et* P' *contient entièrement en son intérieur soit le triangle* $\triangle PQA$, *soit le triangle* $\triangle PQB$, *soit les deux (Figure 11).*

Dém. : Soient A' et B' les autres sommets du carré de diagonale PP' du même côté que A et B respectivement. Puisque $\triangle PP'A'$ et $\triangle PP'B'$ contiennent respectivement $\triangle PQA$ et $\triangle PQB$, il suffit de montrer que le cercle contient soit A', soit B'. Sans perte de généralité, on peut supposer que $P(0, 2b)$ et $P'(2a, 0)$ avec $a \geq b \geq 0$ d'où $A'(a+b, a+b)$. Montrons alors que le centre C du cercle (Γ) est plus proche de A' que de P s'il est situé au-dessus de la droite PP' (l'autre cas se traite de la même manière).

Les points de la médiatrice de $[PP']$ sont à une distance de P supérieure ou égale à $\sqrt{a^2+b^2}$.

- Si $C = R$, A' est sur le périmètre du cercle.
- Si C est situé entre R et A',

$$\left\{\begin{array}{l} a \leq x_c \leq a+b \\ b \leq y_c \leq a+b \end{array}\right. \Longrightarrow d(C, A') = \sqrt{(a+b-x_c)^2 + (a+b-y_c)^2}$$

$$\Longrightarrow d(C, A') \leq \sqrt{a^2+b^2} = d(R, P) \leq d(C, P)$$

- Si C est situé au dessus de A',

$$\left\{\begin{array}{l} x_c \geq a+b \\ y_c \geq a+b \end{array}\right. \Longrightarrow d(C, A') = \sqrt{(x_c-(a+b))^2 + (y_c-(a+b))^2}$$

$$\Longrightarrow d(C, A') \leq \sqrt{x_c^2 + (y_c - 2b)^2} = d(C, P)$$

$\diamond$

Lemme 4.3 *([KK 88]) Supposons que la densité de probabilité f soit quasi-uniforme avec les bornes c_1 et c_2. Soit un rectangle $U_1 \subseteq U$ et $TD(S \cap U_1) < TD(S)$. Si $s \in S \cap U_1$ est un sommet situé à une distance t de la frontière de U_1, alors la probabilité que s soit inachevé dans $TD(S \cap U_1)$ relativement à $TD(S)$ est au plus* $16\left(1 - \frac{c_1\pi}{32}t^2\right)^{N-1}$.

Dém. : Soit C_1 la condition que s est inachevé dans $TD(S \cap U_1)$ relativement à $TD(S)$. Alors, d'après la Proposition 4.1, $C_1 \Rightarrow C_2$, où C_2 est la condition que $TD(S)$ contient une arête (s, q) dont la longueur est plus grande que t. Considérons un cercle de centre s et de rayon $\frac{t}{\sqrt{2}}$. Divisons-le en 16 secteurs ouverts $A_1, \ldots, A_{16}$ d'angles $\frac{\pi}{8}$. Soit C_3 la condition qu'au moins un des secteurs ouverts soit vide (c'est-à-dire n'ait aucun site en son intérieur).

Supposons un instant que $C_2 \Rightarrow C_3$, nous obtenons :

$$P(C_1) \leq P(C_3) \leq \sum_{i=1}^{16} P(S \cap A_i = \emptyset)$$

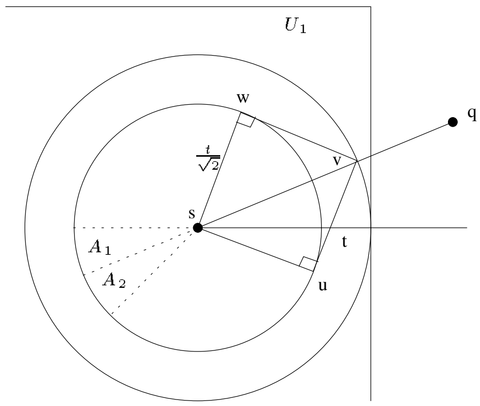

FIG. 12 – *L'un des secteurs A_i ne contient aucun site en son intérieur.*

$$\Longrightarrow P(C_1) = \sum_{i=1}^{16} \left(1 - \int_V f\right)^{N-1} \leq 16\left(1 - \frac{c_1\pi}{32}t^2\right)^{N-1}$$

(nous utilisons la propriété $f(x,y) \geq c_1$).

Il reste à montrer que $C_2 \Rightarrow C_3$. Supposons C_2, et considérons le carré $\square suvw$ avec une diagonale sv s'appuyant sur (s,q), ayant pour extrémité s et une longueur de $\frac{t}{\sqrt{2}}$ (Figure 12). Cette diagonale sv divise le carré en deux triangles $\triangle svu$ et $\triangle svw$. Puisque (s,q) est dans $TD(S)$, il existe un cercle passant par s et q, ne contenant aucun site en son intérieur. Ce cercle contient au moins l'un des deux triangles $\triangle svu$ ou $\triangle svw$ (voir Lemme 4.2). Par conséquent, il contient aussi l'un des secteurs A_i. Par suite, A_i ne contient aucun site en son intérieur. $\diamond$

Corollaire 4.4 *([KK 88]) On se place dans les conditions du Lemme 4.3. Soient a et b les longueurs des côtés de U_1 et $E(a,b)$ l'espérance du nombre de sites inachevés dans U_1. On a :*

$$E(a,b) < 103\left(\frac{c_2}{\sqrt{c_1}}\right)(a+b)\sqrt{N}$$

Dém. : Supposons $a \leq b$. Posons $0 \leq t \leq \frac{a}{2}$. Les points de U_1 situés à une distance de la frontière du rectangle U_1 comprise entre t et $t + dt$ forment une région A_t d'aire $2(a+b-4t)dt$ (Figure 13). La probabilité qu'un site donné appartienne à A_t est :

$$\int_{A_t} f \leq c_2 \times 2(a+b-4t)dt \leq 2c_2(a+b)dt$$

En utilisant le Lemme 4.3, on peut estimer :

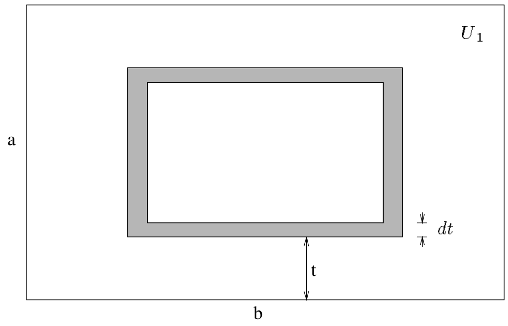

FIG. 13 – *Espérance du nombre de sites inachevés dans le rectangle* U_1.

$$
\begin{aligned}
E(a,b) &\leq N\int_0^{\frac{a}{2}} 16\left(1-\frac{c_1\pi t^2}{32}\right)^{N-1} 2c_2(a+b)dt \\
&= 32c_2(a+b)N\sqrt{\frac{32}{\pi c_1}}\int_0^{a\sqrt{\frac{\pi c_1}{128}}}(1-x^2)^{N-1}dx
\end{aligned}
$$

avec $x = t\sqrt{\frac{\pi c_1}{32}}$

Appelons I l'intégrale. Puisque $a < 1$ et $c_1 \leq 1$,

$$I < \int_0^1 (1-x^2)^{N-1}dx = \int_0^{\frac{\pi}{2}} (\cos\theta)^{2N-1}d\theta = I_{2N-1}$$

avec $x = \sin\theta$.

I est l'intégrale de Wallis qui se calcule par récurrence avec une intégration par parties :

$$
\begin{aligned}
I_{2N-1} &= \frac{2\times 4\times 6\times\cdots(2N-2)}{3\times 5\times 7\times\cdots(2N-1)} \\
2\times I_{2N} &= \frac{3\times 5\times 7\times\cdots(2N-1)}{4\times 6\times 8\times\cdots(2N)}
\end{aligned}
$$

Puisque $\frac{n}{n+1} < \frac{n+1}{n+2}$, on obtient en comparant terme à terme : $I_{2N-1} < 2\times I_{2N}$.

Comme $I_{2N-1}\times(2\times I_{2N}) = \frac{1}{N}$, on déduit :

$$I_{2N-1} < \frac{1}{\sqrt{N}} \Rightarrow I < \frac{1}{\sqrt{N}}$$

d'où $E(a,b) < 103\left(\frac{c_2}{\sqrt{c_1}}\right)(a+b)\sqrt{N}$. $\diamond$

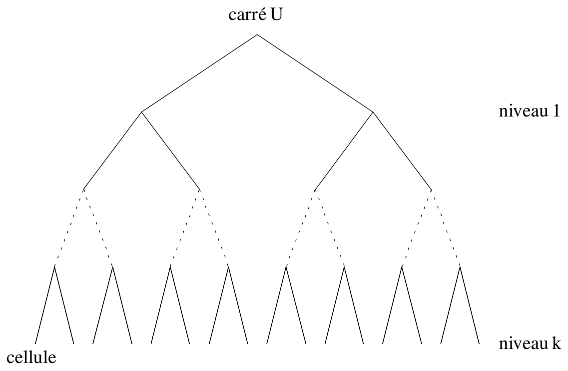

FIG. 14 – *Arborescence schématisant les fusions.*

4.2.4 *Complexité de la triangulation*

Nous allons maintenant utiliser ce résultat pour majorer l'espérance E du nombre total de sites inachevés au cours des différentes fusions. En sommant $(a + b)$ sur toutes les cellules de chaque niveau de fusionnement, on obtient :

$$E < 103 \frac{c_2}{\sqrt{c_1}} \sqrt{N} \sum_{cellules} (a + b)$$

Soit N le nombre de sites situés dans le carré unité U. Comme l'illustre la figure 14, les diverses fusions entre cellules peuvent être schématisées par une arborescence de hauteur $k = \lceil \log_2 N \rceil$. k, qui est la longueur du plus long chemin de cette arborescence, représente aussi le nombre de niveaux de fusionnement.

$$k = \lceil \log_2 N \rceil \Rightarrow 2^{k-1} < N \leq 2^k$$

Remarque : $\sqrt{N} > 2^{\frac{k-1}{2}} \geq 2^{\lceil \frac{k}{2} \rceil - 1}$

Au niveau p de fusionnement, il y a exactement 2^p cellules (sauf pour $p = k$ où il y en a au plus 2^k). Soit N_x le nombre de cellules adjacentes au côté horizontal du carré U et N_y le nombre de cellules adjacentes au côté vertical du carré U. Puisque le processus de division commence verticalement, on a :

$$N_x = 2^{\lceil \frac{p}{2} \rceil} \text{ et } N_y = 2^{\lfloor \frac{p}{2} \rfloor}.$$

On remarque que la somme des côtés horizontaux (*resp.* verticaux) d'une *ligne* (*resp. colonne*) de cellules vaut 1. Comme il y a N_y lignes et N_x colonnes, on obtient en sommant sur toutes les cellules d'un niveau p de fusionnement :

$$\sum_{cellules} (a + b) = N_x + N_y = 2^{\lceil \frac{p}{2} \rceil} + 2^{\lfloor \frac{p}{2} \rfloor} < 2^{\lceil \frac{p}{2} \rceil + 1}$$

Sur l'ensemble des **k niveaux** de fusionnement :

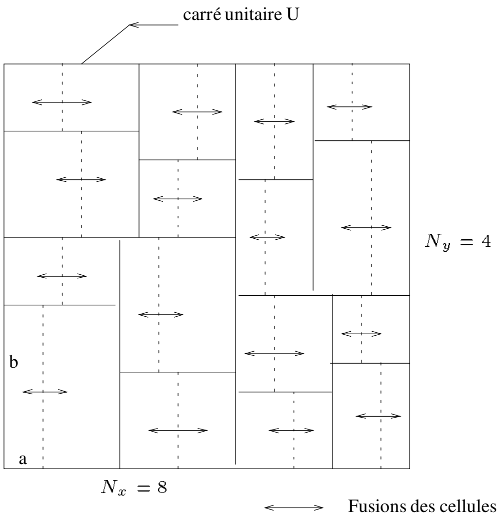

FIG. 15 – *Fusions au niveau p.*

$$\begin{aligned}
\sum_{cellules} (a+b) &< \sum_{p=1}^{k} 2^{\lceil \frac{p}{2} \rceil + 1} \\
&= 2^2 + 2^2 + 2^3 + 2^3 + \cdots + 2^{\lceil \frac{k}{2} \rceil + 1} \\
&\leq 2^3 + 2^4 + 2^5 + \cdots + 2^{\lceil \frac{k}{2} \rceil + 2} \\
&\leq 2^{\lceil \frac{k}{2} \rceil + 3}
\end{aligned}$$

Comme $\sqrt{N} \geq 2^{\lceil \frac{k}{2} \rceil - 1}$, $\sum_{cellules} (a+b) \leq 16\sqrt{N}$,

on obtient donc : $E < 1648 \frac{c_2}{\sqrt{c_1}} N = O(N)$ ce qui constitue un majorant pour l'espérance du nombre total de sites inachevés sur l'ensemble des différentes fusions.

En multipliant par 7 (voir Corollaire 4.3), on obtient un majorant de la complexité en moyenne du processus de triangulation (tri en $2d$ exclu bien sûr). On obtient donc du $O(N)$. $\diamond$

5 Résultats expérimentaux

Expérimentalement, nous avons constaté que le temps de triangulation est divisé d'un facteur variant entre 2 et 3 par rapport à celui de l'algorithme de Lee et Schachter. Lors du processus de fusion, le nombre total d'arêtes créées est voisin de $4N$ (N étant le nombre de sites), le nombre total d'arêtes détruites est voisin de N, et le nombre d'arêtes restant dans la triangulation définitive voisin de $3N$.

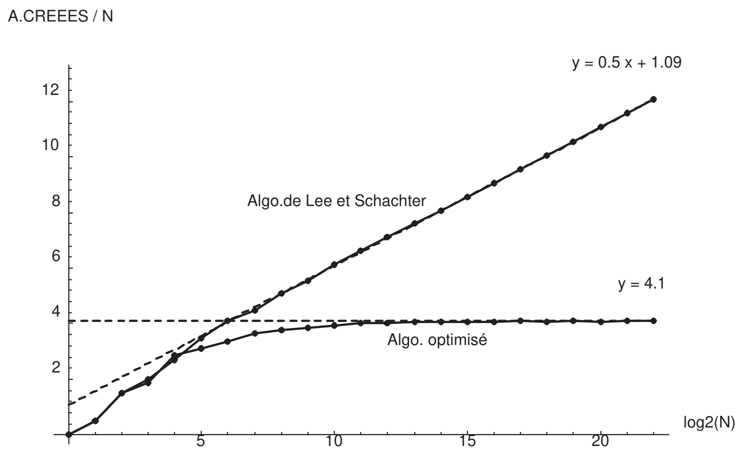

FIG. 16 – *Comparaison entre les deux algorithmes.*

Les tests ont porté sur des ensembles atteignant sept millions de sites répartis dans un domaine carré suivant une loi uniforme. Les algorithmes ont été testés sur une station SUN, sur une Silicon Graphics Indy ainsi que sur un super-calculateur Convex C3, avec lequel les résultats présentés à la Figure 16 ont été obtenus.

Ces résultats "collent" très bien à la théorie et montrent les comportements asymptotiques en $O(N \log N)$ pour l'algorithme de Lee et Schachter et en $O(N)$ pour l'algorithme optimisé (temps consacré au tri exclu). On a choisi de mesurer les performances à l'aide du nombre total d'arêtes créées par le programme puisque le temps de triangulation est proportionnel à ce nombre (voir Corollaire 4.2). On s'affranchit ainsi de la vitesse du processeur utilisé, des problèmes dus au réseau et aux utilisateurs travaillant simultanément sur la même machine. L'écart entre les deux algorithmes commence à être sensible à partir de 60 sites et, pour 130 000 sites, l'algorithme optimisé est déjà deux fois plus rapide. L'écart s'amplifie ensuite au fur et à mesure que N augmente.

Sur une Silicon Graphics Indy (200 Mhz, 64 MO de mémoire vive), on triangule 200 000 sites en 7 secondes après une phase de tri de trois secondes. La vitesse de triangulation (en excluant le temps consacré au tri) est de l'ordre de 30 000 sites (ou 50 000 triangles) par seconde.

6 Remarques et conclusion

Nous avons présenté un algorithme qui, après un tri selon deux directions de N sites répartis de manière quasi-uniforme dans un carré unité, réalise la triangulation de Delaunay en $O(N)$ *en moyenne*.

L'efficacité de cet algorithme est due au fait que les fusions, réalisées alternativement selon le sens vertical et horizontal, mettent relativement moins souvent en cause les triangles déjà construits que dans l'algorithme classique de Lee et Schachter.

Cet algorithme semble avoir le même comportement sur beaucoup d'autres types de distribution (que quasi-uniforme), mais une étude théorique et expérimentale approfondie est nécessaire pour le prouver.

Certains algorithmes réalisent la triangulation de Delaunay en $O(N \log\log N)$ *en moyenne* ([DWY 87]), et même en $O(N)$ *en moyenne* ([KK 88]). Mais ils utilisent tous une grille régulière pour pré-partitionner les points, ce qui réduit considérablement les possibilités d'utilisation de tels algorithmes.

Nous terminons cette étude par le problème ouvert suivant, déjà mentionné par O. Devillers (*INRIA*, Sophia Antipolis) : *est-il possible, à partir de la connaissance du tri selon deux directions d'un ensemble de N sites du plan, de calculer leur triangulation de Delaunay euclidienne en $o(N \log N)$ dans le pire des cas?*

Note : Signalons, pour finir, les travaux de J.C. Spehner de l'Université de Haute Alsace qui calcule le diagramme de Delaunay en utilisant des mélanges de tris [AES 96].

Remerciements : Les auteurs tiennent à remercier, au *SETRA*, Jacques Hervé pour son aide en informatique et ses encouragements, Philippe Nouaille pour sa participation au codage de l'optimisation, Tam Vo-Dinh pour la mise à disposition du calculateur Convex C3, et enfin Jean-Noël Theillout, qui a permis à C. Lemaire de mener à terme cette étude.

Références

[AES 96] B. ADAM, M. ELBAZ, J.C. SPEHNER. Construction du diagramme de Delaunay dans le plan en utilisant les mélanges de tris. *Journées de l'AFIG 96*, Dijon, nov. 1996.

[AUR 91] F. AURENHAMMER. Voronoi diagrams: a survey of a fundamental geometric data structure, *ACM Comput. Surv.*, vol 23, pages 345–405, 1991.

[BAU 75] B.G. BAUMGART. A polyhedron representation for Computer Vision. *National Computer Conference*, pages 589–596, 1975.

[BEN 75] J.L. BENTLEY. Multidimensional binary search trees used for associated searching. *Communications of the ACM*, vol. 18, pages 509–517, 1975.

[BFPRT 73] M. BLUM, R.W. FLOYD, V. PRATT, R.L. RIVEST, R.E. TARJAN. Time bounds for selection. *Journal of Computer and System Sciences*, 7(4), pages 448–461, 1973.

[CLR 94] T.H. CORMEN, C.E. LEISERSON, R.L. RIVEST. *Introduction à l'algorithmique*. Traduction de X. Cazin, Dunod, Paris, 1994.

[COR 75] R. CORI. Un code pour les graphes planaires et ses applications. *Astérisque*, 27, 1975.

[DWY 87] R.A. DWYER. A faster divide-and-conquer algorithm for constructing Delaunay triangulations. *Algorithmica*, 2, pages 137–151, 1987.

[DWY 91] R.A. DWYER. Higher-Dimensional Voronoi Diagrams in Linear Expected Time. *Discrete Comput. Geom.*, 6, pages 343–367, 1991.

[GS 85] L.J. GUIBAS, J. STOLFI. Primitives for the Manipulation of General Subdivisions and the Computation of Voronoi Diagrams. *ACM Transactions on Graphics*, 4, pages 74–123, 1985.

[HOU 87] P. HOUTHUYS. Box Sort, a multidimensional binary sorting method for rectangular boxes, used for quick range searching. *The Visual Computer*, 3, pages 236–249, 1987.

[KK 88] J. KATAJAINEN, M. KOPPINEN. Constructing Delaunay triangulations by merging buckets in quadtree order. *Annales Societatis mathematicae Polonae*, Series IV, Fundamenta Informaticæ, 11, pages 275–288, 1988.

[LS 80] D.T. LEE, B.J. SCHACHTER. Two algorithms for constructing a Delaunay triangulation. *International Journal of Computer and Information Sciences*, 9, pages 219–242, 1980.

[MAU 84] A. MAUS. Delaunay triangulation and the convex hull of n points in expected linear time. *BIT*, 24, pages 151–163, 1984.

[OIM 84] T. OHA, M. ITI, K. MUROTA. Improvements of the incremental method for the Voronoi diagram with computational comparison of various algorithms. *Journal of the Operations Research Society of Japan*, 27, pages 306–337, 1984.

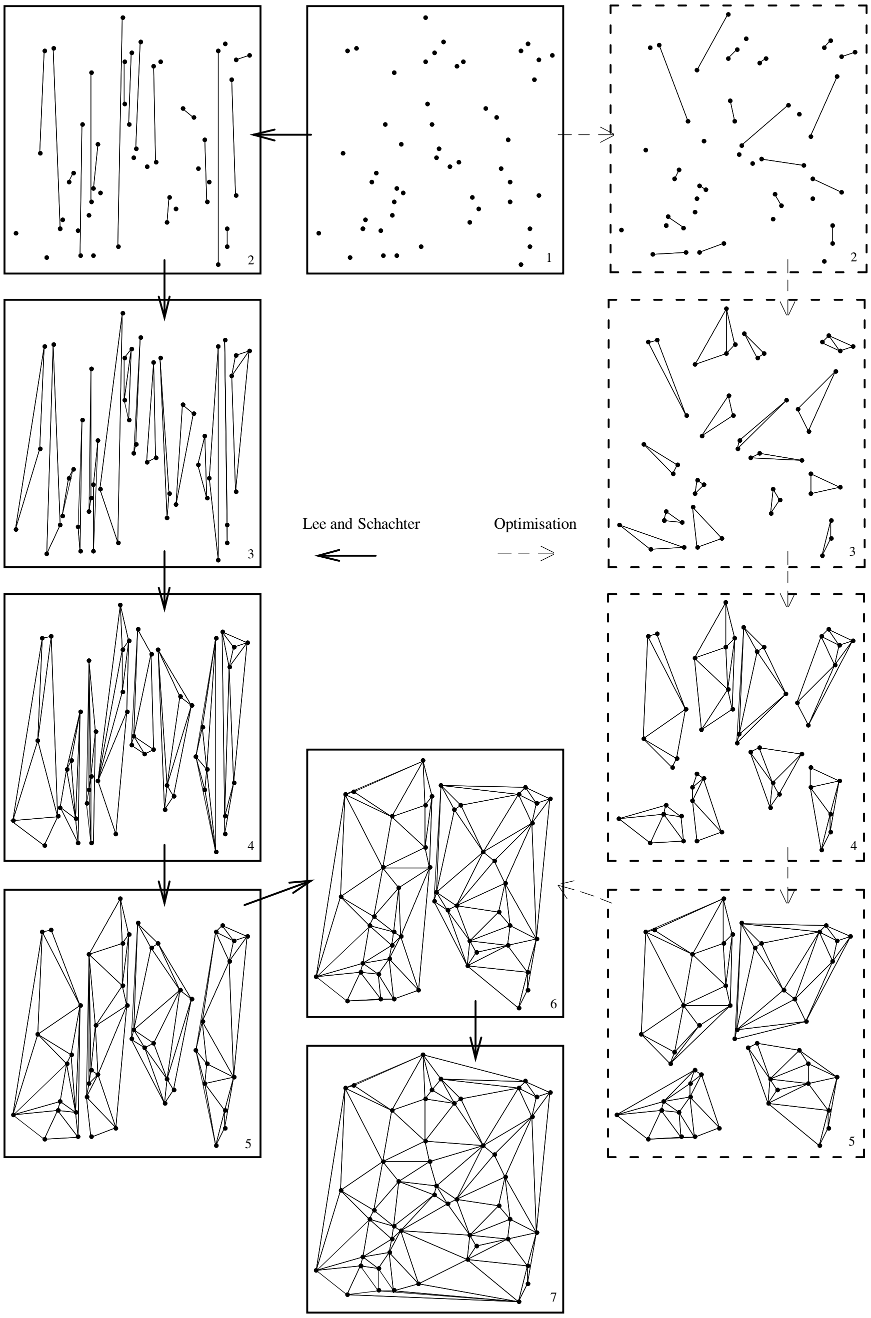

FIG. 17 – *Etapes successives de la triangulation : comparaison entre les deux algorithmes (ensemble de* 50 *sites).*