Sudokus et algorithmes de recuit

par Renaud Sirdey*

Nous présentons un algorithme de recuit simulé pour une application ludique d'actualité : la résolution de Sudokus. Dans un prochain numéro, nous présenterons une méthode basée sur la programmation linéaire.

Introduction

Le Sudoku (est-il encore besoin de le présenter ?) semble être *le* casse-tête du moment. Dans la mesure où il cache un problème combinatoire redoutable, il est fort tentant de s'en servir pour illustrer, de manière ludique, l'art de concevoir des algorithmes de résolution de problèmes combinatoires. C'est ce que nous faisons ici avec la méthode dite du recuit simulé.

I Règles du jeu

Un Sudoku consiste en la donnée d'une grille carrée de 81 cases divisée en 9 *sous-carrés* de 3 cases de côté et partiellement remplie avec des chiffres de 1 à 9. Le but du jeu est de compléter le remplissage de la grille avec des chiffres de 1 à 9 de manière à ce qu'aucun chiffre n'apparaisse deux fois dans la même ligne, dans la même colonne ou dans le même sous-carré. En règle générale, il n'y a qu'une seule façon de compléter la grille.

Le tableau I donne un exemple de grille complétée.

Comme le fait remarquer Jean-Paul Delahaye [3], compléter un Sudoku est équivalent à compléter le 9-coloriage d'un graphe $G = (V, E)$ dont les sommets sont les cases de la grille et où deux sommets sont reliés par une arête s'ils appartiennent à la même ligne, à la même colonne ou au même sous-carré. Rappelons qu'un 9-coloriage est une application $f : V \mapsto \{1, 9\}$ telle que $f(v) \neq f(w)$ dès lors que $\{v, w\} \in E$.

Notons aussi que le problème qui consiste à décider si un Sudoku (de taille arbitraire) peut être complété ou non est *NP*-complet [17]. Ceci justifie une approche heuristique, si tant est, bien évidemment, que les algorithmes résultants soient polynomiaux.

* renauds@nortel.com

Tableau I. Exemple de Sudoku « diabolique » complet. Les chiffres encadrés indiquent les données du problème.

[6]	5	9	4	1	[3]	[7]	[8]	2
3	[1]	7	2	8	[5]	4	[9]	6
2	4	[8]	7	9	6	5	[3]	[1]
9	6	5	[1]	7	2	8	4	3
8	7	[2]	9	[3]	4	1	[6]	5
4	3	1	5	6	[8]	2	7	[9]
[7]	2	6	3	4	1	9	5	8
5	8	4	[6]	[2]	[9]	3	1	7
[1]	[9]	3	8	5	7	[6]	2	4

II Le recuit simulé

La méthode du recuit simulé a été proposée dans les années 80 [2, 9] et, à l'origine, justifiée par analogie avec la technique dite du recuit utilisée par les physiciens afin de conduire certains systèmes physiques dans un état de basse énergie. Il s'agit d'une *métaheuristique* [4], autrement dit d'un principe général de construction d'algorithmes de résolution approchée pour les problèmes d'optimisation difficile, en particulier combinatoire. Ses avantages sont d'être relativement bien comprise sur le plan théorique et de conduire à des algorithmes assez simples. Elle a par ailleurs été utilisée avec succès dans le cadre d'applications industrielles (voir [14] pour un exemple), et ce à de nombreuses reprises.

Soit un problème d'optimisation combinatoire qui consiste, étant donné un ensemble de solutions $\Omega = \{\omega_1, \ldots, \omega_N\}$ ainsi qu'une *fonction économique* $c : \Omega \mapsto \{e_1, \ldots, e_P\}$, à trouver un élément $\omega^\star \in \Omega$ tel que $e_1 = c(\omega^\star) \leq c(\omega) \leq e_P$, pour tout $\omega \in \Omega$. De manière usuelle, $c(\omega)$ s'appelle la *valeur* de la solution ω. Aussi, on suppose donnée une *fonction de voisinage* $V : \Omega \mapsto 2^\Omega$.

La méthode est alors très simple à énoncer. Au départ on commence avec une solution arbitraire. Soit ω la solution courante. À chaque itération, une solution candidate ω' est choisie uniformément dans le voisinage de la solution courante et acceptée avec une probabilité égale à $\min\left(1, \mathrm{e}^{-\frac{c(\omega')-c(\omega)}{T}}\right)$. Le paramètre T s'appelle la *température* et tend vers 0, de manière monotone, selon une fonction appelée *loi de décroissance de la température*. La meilleure solution rencontrée durant l'exécution de l'algorithme est mémorisée et écrite lorsque la condition d'arrêt choisie est vérifiée.

Bien que ce schéma converge en probabilité vers une solution optimale sous des conditions assez peu restrictives (symétrie de la fonction de voisinage *i.e.* $\omega_j \in V(\omega_i) \Leftrightarrow \omega_i \in V(\omega_j)$ et forte connexité du graphe orienté induit), ce n'est qu'au prix d'une décroissance de la température extrêmement (comprendre prohibitivement) lente [6]. Qui plus est, on sait qu'il existe des instances du problème du couplage de cardinalité maximum (problème polynomial) et du problème de 3-coloriage d'un graphe dont la résolution nécessite un nombre exponentiel d'itérations [11, 13].

Ces résultats négatifs ne doivent pas obscurcir le tableau outre mesure : la pertinence pratique de la méthode reste remarquable !

Généralement, la mise en œuvre d'un algorithme de recuit simulé passe par l'étude de la chaîne de Markov qui modélise l'algorithme lorsque la température reste constante et dont la *matrice des probabilités de passage* est donnée par [1, 10, 16] :

$$A_{ij}(T) = \begin{cases} 0 & \text{si } \omega_j \notin V(\omega_i) \\ \frac{1}{|V(\omega_i)|} & \text{si } \omega_j \in V(\omega_i) \text{ et } c(\omega_j) \leq c(\omega_i) \\ \frac{\mathrm{e}^{\frac{c(\omega_i)-c(\omega_j)}{T}}}{|V(\omega_i)|} & \text{si } \omega_j \in V(\omega_i) \text{ et } c(\omega_j) > c(\omega_i) \\ 1 - \sum_{j:\omega_j \in V(\omega_i)} A_{ij}(T) & \text{si } i = j \end{cases}.$$

Sous les hypothèses de *régularité* (c'est-à-dire, dans le présent contexte, de forte connexité du graphe orienté induit par la fonction de voisinage) et d'*apériodicité*, cette chaîne admet une unique *loi stationnaire* exprimée comme suit

$$\pi_i^{(\infty)}(T) = \frac{\mathrm{e}^{-\frac{c(\omega_i)}{T}}}{\sum_{j=1}^{N} \mathrm{e}^{-\frac{c(\omega_j)}{T}}}. \qquad (1)$$

Aussi,

$$\lim_{T\to 0} \pi_i^{(\infty)}(T) = \lim_{T\to 0} \frac{1}{\sum_{j=1}^{N} \mathrm{e}^{\frac{c(\omega_i)-c(\omega_j)}{T}}} = \begin{cases} 0 & \text{si } c(\omega_i) > e_1 \\ \frac{1}{|\Omega^\star|} & \text{sinon} \end{cases},$$

où $\Omega^\star = \{\omega \in \Omega : c(\omega) = e_1\}$.

III Résolution de Sudokus

Nous l'avons vu, la méthode du recuit simulé permet de s'attaquer à des problèmes d'optimisation (ici, combinatoire). Pour l'utiliser afin de résoudre des Sudokus, il convient d'associer un problème d'optimisation à chaque instance. Pour ce faire, nous procédons de la manière suivante : l'ensemble des solutions, Ω, est l'ensemble des grilles carrées de côté 9 dont les cases contiennent un chiffre de 1 à 9 quelconque, ceci à l'exception des cases *fixes* qui définissent l'instance. S'il y a m telles cases, on a $N = |\Omega| = 9^{81-m}$. Soit $\omega \in \Omega$, ω_{ij} dénotant le contenu de la case dont la ligne est i et la colonne j. On note $c_{ij}(\omega)$ le nombre d'occurences du chiffre ω_{ij} dans la zone définie par l'union de la ligne i, de la colonne j et du sous-carré contenant la case ij, cette dernière n'étant pas comprise ; la fonction économique est alors définie par

$$c(\omega) = \frac{1}{2} \sum_{i=1}^{9} \sum_{j=1}^{9} c_{ij}(\omega).$$

Il est clair qu'une solution de valeur nulle respecte les règles du jeu et résout donc le Sudoku.

La fonction de voisinage, quant à elle, est définie comme suit. Deux grilles sont voisines l'une de l'autre si l'une peut s'obtenir à partir de l'autre en changeant le chiffre contenu dans une et une seule case (non fixe bien sûr). La forte connexité du graphe orienté induit par cette relation de voisinage est évidente.

Il convient ensuite de chercher à simuler avec une précision acceptable la loi stationnaire à une température suffisamment faible pour avoir de bonnes chances de tirer une solution optimale. Pour ce faire, l'idée consiste à démarrer l'algorithme à une température relativement élevée, température à laquelle la convergence vers la loi stationnaire est extrêmement rapide[1], et à faire décroître la température de telle manière que

[1] Rappelons que la convergence vers la loi stationnaire est géométrique et que la vitesse de convergence dépend de la valeur absolue de la deuxième plus grande valeur propre de la matrice des probabilités de passage [8, 12]. Malheureusement, celle-ci est extrêmement faible à basse température.

les lois stationnaires associées à deux valeurs successives soient proches. Typiquement [1], on choisira

$$T_{k+1} = \frac{T_k}{1 + \frac{\log(1+\delta)}{e_P+1} T_k}, \tag{2}$$

ce qui garantit que

$$|\pi_i^{(\infty)}(T_k) - \pi_i^{(\infty)}(T_{k+1})| \leq \delta,$$

où δ est un petit réel positif. En conséquence, il est raisonnable d'escompter qu'après avoir fait décroître la température il suffit d'un petit nombre d'itérations pour se rapprocher de la nouvelle loi stationnaire, de manière satisfaisante. Bien entendu, ce raisonnement est de nature heuristique et il existe d'autres façons de procéder [15].

Reste à fixer le nombre d'itérations par palier et à savoir quand s'arrêter lorsque l'algorithme tarde à fournir une solution de valeur nulle. Concernant le premier point, on choisit généralement un nombre d'itérations proportionnel à la taille « naturelle » du problème, ici le nombre de cases de la grille. Concernant le second point, on sait [10], pour un problème d'optimisation combinatoire quelconque, que les solutions issues de la loi stationnaire à la température

$$T_f = \frac{\varepsilon}{\log N - \log(1-\alpha)} \tag{3}$$

ont une valeur inférieure ou égale à $e_1 + \varepsilon$ avec une probabilité d'au moins α. Afin de s'en convaincre, il suffit de remarquer que

$$\begin{aligned}
1 - \alpha &= \sum_{i:e_i > e_1+\varepsilon} \frac{N(i)\mathrm{e}^{-\frac{e_i}{T}}}{K(T)} \\
&\leq \frac{\mathrm{e}^{-\frac{e_1+\varepsilon}{T}}}{K(T)} \underbrace{\sum_{i:e_i>e_1+\varepsilon} N(i)}_{\leq N} \\
&\leq N e^{-\frac{\varepsilon}{T}} \underbrace{\frac{\mathrm{e}^{-\frac{e_1}{T}}}{K(T)}}_{\leq 1} \\
&\leq N e^{-\frac{\varepsilon}{T}},
\end{aligned}$$

où $N(i)$ est le nombre de solutions dont la valeur est égale à e_i et où

$$K(T) = \sum_{j=1}^{N} \mathrm{e}^{-\frac{c(\omega_j)}{T}}.$$

Pour le présent problème, on choisira, par exemple, $\varepsilon = \frac{1}{2}$, dans la mesure où la fonction économique est à valeurs entières. L'équation (3) se réécrit alors

$$T_f = \frac{0.5}{(81-m)\log 9 - \log(1-\alpha)}.$$

En pratique, on pourra lui préférer la valeur

$$T'_f = \frac{0.5}{81\log 9 - \log(1-\alpha)}$$

qui fournit les mêmes garanties tout en ayant l'avantage d'être indépendante de l'instance considérée.

Par exemple, si l'algorithme atteint la température 0,002 738 52 il est raisonnable de conclure qu'une proximité suffisante à la loi stationnaire n'a pas pu être maintenue car, dans le cas contraire, il y avait plus de 99 % de chances de tirer une solution de valeur nulle. Il convient alors d'abandonner et, par exemple, de relancer l'algorithme dans l'espoir de faire mieux.

L'encadré suivant donne l'énoncé de l'algorithme.

Énoncé de l'algorithme

Poser $\delta = 0.1$, $e_P = \frac{1620}{2}$ et faire $T \leftarrow e_P$.

Choisir ω arbitrairement et faire $c \leftarrow c(\omega)$.

Tant que $T \geq 0.00273852$ faire

  Palier : pour $k = 1$ à 81 faire

    Choisir i et j uniformément dans $\{1, 9\}$ tant que la case ij est fixe.

    Faire $t \leftarrow \omega_{ij}$ et $c_1 \leftarrow c_{ij}(\omega)$.

    Choisir ω_{ij} uniformément dans $\{1, 9\}$ tant que $\omega_{ij} = t$.

    Faire $c_2 \leftarrow c_{ij}(\omega)$ et $c' \leftarrow c - c_1 + c_2$.

    Choisir u uniformément dans $[0, 1]$.

    Si $u \leq e^{-\frac{c'-c}{T}}$ alors faire $c \leftarrow c'$ (acceptation).

    Sinon faire $\omega_{ij} \leftarrow t$ (rejet).

    Si $c = 0$ alors écrire ω et s'arrêter.

  Fin.

  Faire $T \leftarrow \frac{T}{1+\frac{\log(1+\delta)}{e_P+1}T}$.

Fin.

IV Résultats empiriques

Pour fixer les idées, nous avons essayé l'algorithme sur quatre Sudokus de niveau « diabolique » donnés dans les tableaux I, II, III et IV.

Ces quatres Sudokus ont respectivement 23, 26, 25 et 24 cases fixes et il faut, en moyenne[2], respectivement 7,69, 2,28, 3,85 et 2,38 essais[3] pour en venir à bout. Notons que l'algorithme ne parvient à résoudre le Sudoku « diabolique » donné dans [3] (23 cases fixes) qu'au prix de 11,11 essais, en moyenne.

[2] Estimée sur 100 essais.

[3] Un essai requiert de l'ordre de trois secondes sur un ordinateur portable des plus moyens.

Tableau II. Un Sudoku de niveau « diabolique ».

5	4	9	1	7	2	3	8	6
6	8	3	4	5	9	1	7	2
2	7	1	3	8	6	5	9	4
3	2	4	5	9	7	6	1	8
7	6	5	8	2	1	4	3	9
9	1	8	6	4	3	2	5	7
1	9	7	2	6	5	8	4	3
4	5	6	9	3	8	7	2	1
8	3	2	7	1	4	9	6	5

Tableau III. Un autre Sudoku de niveau « diabolique ».

9	5	2	1	4	7	6	3	8
1	7	6	8	9	3	2	4	5
4	8	3	2	5	6	7	1	9
6	1	4	7	3	8	9	5	2
7	2	9	5	6	1	4	8	3
5	3	8	4	2	9	1	7	6
3	4	5	9	7	2	8	6	1
2	6	1	3	8	4	5	9	7
8	9	7	6	1	5	3	2	4

Tableau IV. Encore un autre Sudoku de niveau « diabolique ».

7	9	8	6	4	2	3	1	5
4	1	5	9	7	3	6	2	8
3	6	2	1	8	5	9	7	4
6	7	4	5	2	8	1	9	3
5	2	1	7	3	9	4	8	6
8	3	9	4	6	1	7	5	2
9	4	3	8	5	7	2	6	1
1	8	6	2	9	4	5	3	7
2	5	7	3	1	6	8	4	9

Aussi, précisons que lorsque l'algorithme n'arrive pas à résoudre un Sudoku, il termine généralement avec une solution de valeur très faible (typiquement 2) ce qui est excellent sur le plan de l'optimisation, bien qu'inutile pour le Sudoku (rappelons toutefois qu'il n'y a théoriquement qu'une et une seule solution de valeur nulle, ce qui complique la tâche de l'algorithme).

Enfin, il convient d'insister sur le fait que nous ne prétendons ni que notre algorithme est compétitif avec les meilleurs algorithmes de résolution de Sudokus ni qu'il n'existe pas de manière plus ingénieuse d'appliquer la méthode du recuit simulé à ce problème (en particulier, de nombreux travaux traitent de l'application de la méthode au problème du coloriage, par exemple [7], et, à ce titre, sont d'un intérêt certain pour le présent problème).

V Généralisation et complexité

Le Sudoku, tout comme notre algorithme, se généralise aisément. Soit n le côté des sous-carrés. Un n-Sudoku consiste en la donnée d'une grille carrée de n^4 cases divisée en n^2 sous-carrés de n cases de côté et partiellement remplie avec des nombres de 1 à n^2 (rappelons que le nombre chromatique d'un graphe G est supérieur ou égal à la cardinalité maximale d'une clique de G [5], or, pour notre problème, un sous-carré définit une clique de cardinalité n^2 contenue dans le graphe à colorier).

Considérons que T_0, α, δ et ε sont fixés et commençons par quantifier le nombre de paliers de température rencontrés par l'algorithme. Il est aisé de vérifier que la loi de décroissance de la température donnée par l'équation (2) est telle que

$$T_{k+l} = \frac{T_k}{1 + l\frac{\log(1+\delta)}{e_P+1}T_k}.$$

Il suit que le nombre de paliers de température rencontrés, noté Λ, est la solution de l'équation

$$\frac{T_0}{1 + \Lambda\frac{\log(1+\delta)}{e_P+1}T_0} = \frac{\varepsilon}{\log N - \log(1-\alpha)},$$

soit

$$\Lambda = \frac{(1+e_P)(\log N - \log(1-\alpha))}{\varepsilon \log(1+\delta)} - \frac{1+e_P}{T_0 \log(1+\delta)}.$$

Notons qu'avec $T_0 = e_P$ le second terme de l'équation ci-dessus est approximativement égal à $\frac{1}{\log(1+\delta)}$, dès que e_P est suffisamment grand.

Pour un n-Sudoku, e_P est en $O(n^6)$, le nombre de paliers rencontrés est donc en $O(n^{10}\log n)$, dans la mesure où $N \leq n^{2n^4}$. Puisque l'on réalise n^4 itérations pour chaque valeur de la température et que le calcul de la valeur de la solution candidate est en $O(n^2)$, la complexité de l'algorithme est en $O(n^{16}\log n)$, donc polynomiale. Toutefois, la taille « naturelle » du problème est plutôt le nombre de cases de la grille. Soit $p = n^4$, l'algorithme est alors en $O(p^4 \log p)$.

Ce dernier résultat ne doit néanmoins pas masquer le fait que l'algorithme, bien qu'à complexité polynomiale, reste relativement « gourmand » : les constantes cachées dans le O sont loin d'être négligeables. Afin d'améliorer l'efficacité pratique de la méthode, on pourra ajouter une condition d'arrêt *ad hoc* consistant à abandonner dès lors que la valeur de la meilleure solution rencontrée ne s'est pas améliorée durant un nombre suffisamment grand (typiquement 10 000) de paliers de température.

Références

[1] E.H.L. Aarts et P.J.M. van Laarhoven, « Statistical cooling : a general approach to combinatorial optimization problems », *Philips J. Res.* **40** (4) (1985) 193–226.

[2] V. Cerny, « Thermodynamical approach to the traveling salesman problem : an efficient simulation algorithm », *J. Optimiz. Theory App.* **5** (1) (1985) 41–51.

[3] J.-P. Delahaye, « Le tsunami du Sudoku », *Pour la Science* (décembre 2005).

[4] J. Dréo, A. Pétrowski, P. Siarry et É. Taillard, *Métaheuristiques pour l'optimisation difficile*, Algorithmes, Eyrolles, 2003.

[5] M. Gondran et M. Minoux, *Graphes et algorithmes*, tome 37 de *Collection de la Direction des Études et Recherches d'Électricité de France*, Eyrolles, troisième édition, 1995.

[6] B. Hajek, « Cooling schedule for optimal annealing », *Math. Oper. Res.* **13** (1988) 311–329.

[7] D.S. Johnson, C.R. Aragon, L.A. McGeogh et C. Schevon, « Optimization by simulated annealing an experimental evaluation part. II : graph coloring and number partitioning », *Oper. Res.* **39** (1991) 378–406.

[8] J.G. Kemeny et J.L. Snell, *Finite Markov Chains*, The University Series in Undergraduate Mathematics, D. van Nostrand Company, Princeton, New Jersey, 1960.

[9] S. Kirkpatrick, C.D. Gelatt Jr et M.P. Vecchi, « Optimization by simulated annealing », *Science* (1983).

[10] M. Lundy et A. Mees, « Convergence of an annealing algorithm », *Math. Program.* **34** (1986) 111–124.

[11] A. Nolte et R. Schrader, « Simulated annealing and its problems to color graphs », In *Algorithms – ESA 96*, tome 1136 de *Lect. Notes Comput. Sci.* Springer, 1996, pp. 138–151.

[12] S. Ross, *Stochastic Processes*, Probability and Statistics, John Wiley & Sons, deuxième édition, 1996.

[13] G.H. Sasaki et B. Hajek, « The time complexity of maximum matching by simulated annealing », *J. ACM* **35** (2) (1988) 387–403.

[14] R. Sirdey, J. Carlier et D. Nace, « Approximate resolution of a resource-constrained scheduling problem », Rapport technique PE/BSC/INF/016550 V01 EN, Service d'architecture BSC, Nortel GSM Access R&D, France (soumis au Journal of Heuristics), 2005.

[15] E. Triki, Y. Colette et P. Siarry, « A theoretical study on the behavior of simulated annealing leading to a new cooling schedule », *Eur. J. Oper. Res.* **166** (2005) 77–92.

[16] P.J.M. van Laarhoven et E.H.L. Aarts, *Simulated annealing : theory and applications*, Mathematics and its Applications, Kluwer Academic Publisher, 1987.

[17] T. Yato, *Complexity and completeness of finding another solution and its application to puzzles*, MSc thesis, Université de Tokyo, 2003.