Rapport DIUL-RT-8803, 1988

BonGCL

Un logiciel pour la recherche de bons générateurs à congruence linéaire [1]

Pierre L'Ecuyer et François Blouin

Département d'informatique
Université Laval

Résumé

BonGCL est un logiciel qui permet de chercher des générateurs à congruence linéaire (GCL) d'ordre k de période maximale et de leur appliquer le test spectral. Il produit un rapport qui contient les générateurs ayant le mieux réussi ce test parmi ceux examinés, leurs résultats, ainsi que des statistiques sur la recherche effectuée.

[1] La mise au point de ce logiciel a bénéficié de subventions de recherche du CRSNG (Canada) et du FCAR (Québec) au premier auteur.

1 Introduction

BonGCL est un logiciel écrit en Modula-2 qui permet de chercher des générateurs à congruence linéaire (GCL) d'ordre k de période maximale et de leur appliquer le test spectral. Il produit un rapport qui contient les générateurs ayant le mieux réussi ce test parmi ceux examinés, leurs résultats, ainsi que des statistiques sur la recherche effectuée. Le lecteur trouvera des explications détaillées sur les GCL d'ordre k et le test spectral dans le document [2] pour lequel ce logiciel a été développé.

Les GCL d'ordre k sont des récursions de la forme

$$X_n := a_1 X_{n-1} + \cdots + a_k X_{n-k} \bmod m$$

qui peuvent servir de générateurs de nombres pseudo-aléatoires sur les ordinateurs. Ils ont de bonnes propriétés statistiques lorsqu'ils sont bien choisis, ils sont faciles à initialiser et leur implantation est relativement efficace. De plus, leur période peut atteindre $m^k - 1$ si les multiplicateurs $a_1, \ldots, a_k$ et le modulo m respectent certaines conditions. Pour un m et un k donnés, la proportion de vecteurs $a = (a_1, \ldots, a_k)$ respectant ces conditions est assez importante pour qu'il soit facile d'en trouver en cherchant au hasard.

Le test spectral est un test théorique puissant qui porte un jugement sur l'ensemble de la séquence sans la générer [1]. Il permet d'examiner la structure que forment les t-tuples de valeurs successives de X_n/m dans l'hypercube $(0,1)^t$. Ces points se regroupent sur des familles d'hyperplans parallèles équidistants. Le test spectral trouve la famille pour laquelle la distance entre deux hyperplans adjacents est maximale et calcule cette distance, notée $d_t(k, m, a)$. Plus elle est petite, plus les tranches vides dans l'hypercube sont petites, ce qui assure habituellement de bonnes propriétés au générateur. La valeur de $d_t(k, m, a)$ ne peut être inférieure à une certaine borne théorique connue, $d_t^*(k, m)$. Pour obtenir une valeur entre 0 et 1, on pose

$$S_t(k, m, a) = \frac{d_t^*(k, m)}{d_t(k, m, a)}.$$

Plus cette valeur est élevée, meilleur est le résultat du GCL au test pour t dimensions. Un bon générateur devrait obtenir un résultat satisfaisant pour chaque valeur de t de 1 à T, où T est la plus grande dimension pour laquelle on veut appliquer le test. Pour $t \leq k$, la valeur de d_t ne dépend que de m et n'est donc pas utile pour comparer différents vecteurs de multiplicateurs. Le logiciel mesure le mérite d'un générateur en calculant

$$M_T(k, m, a) \stackrel{\text{def}}{=} \min_{k<t\leq T} S_t(k, m, a).$$

BonGCL est présentement implanté sous VAX/VMS et sur le Macintosh de Apple (sous MPW). Il utilise le logiciel SENTIERS [3] pour représenter et manipuler les entiers de grande taille sans perte de précision.

2 Description de la méthode de recherche

Pour un m et un k donnés, le logiciel effectue une recherche à l'intérieur d'une région spécifique bornée par les vecteurs $b = (b_1, \ldots, b_k)$ et $c = (c_1, \ldots, c_k)$ tels que $-m < b_i \leq c_i < m,\ \forall i$. Cette recherche peut être soit exhaustive, soit aléatoire.

1. Si elle est exhaustive, tous les vecteurs $a = (a_1, \ldots, a_k)$ tels que $b_i \leq a_i \leq c_i,\ i = 1, \ldots, k$ sont examinés.
   Le nombre total de GCL examinés sera donc $\prod_{i=1}^{k}(c_i - b_i + 1)$.

2. Si elle est aléatoire, alors le programme doit connaître n, le nombre de sous-régions à examiner, et h_i, la taille d'une sous-région selon l'axe i, pour $i = 1, \ldots, k$. Le programme répète n fois :

   2a. Pour $i = 1, \ldots, k$, il génère α_i selon une loi uniforme discrète dans l'ensemble des entiers $\{b_i, \ldots, c_i - h_i + 1\}$. Il examine chacun des vecteurs $(a_1, \ldots, a_k)$ tels que $\alpha_i \leq a_i \leq \alpha_i + h_i - 1,\ \ i = 1, \ldots, k$.

   Le nombre total de GCL examinés sera donc $n \prod_{i=1}^{k}(h_i)$.

Pour examiner un vecteur a, le logiciel vérifie d'abord s'il respecte les conditions prouvant que le générateur correspondant est de période maximale. La condition (a) du Théorème 2 (voir [2], section 2.3) n'est vérifiée qu'une seule fois pour chaque valeur distincte de a_k, ce qui est avantageux lorsque cette valeur est la même pour plusieurs vecteurs a. La vérification des conditions de période maximale nécessite la factorisation de $(m-1)$ et de $r = (m^k - 1)/(m-1)$. Si l'usager le désire, le programme pourra trouver lui-même ces factorisations, sinon elles devront être fournies dans un fichier (voir section suivante). Si le GCL a une période maximale, il est soumis au test spectral pour les dimensions $t = k+1, \ldots, T$, sinon il est rejeté. Pour apparaître dans le rapport final, un générateur doit obtenir une valeur de M_T supérieure ou égale à *SeuilRejet* (une valeur fournie par l'usager) et qui est parmi les C plus grandes valeurs de M_T des générateurs de période maximale examinés. Dès qu'un résultat pour une dimension donnée permet de savoir qu'un générateur n'apparaîtra pas dans le rapport final, on le rejette sans poursuivre inutilement les calculs.

3 Instructions pour l'utilisation

Les données à être fournies au programme doivent être placées dans un fichier dont l'extension est "`.DAT`", en respectant le format de la figure 1. Les données entre crochets sont optionnelles; leur présence dépend de la valeur du champ en début de ligne. La signification des champs est expliquée ci-bas.

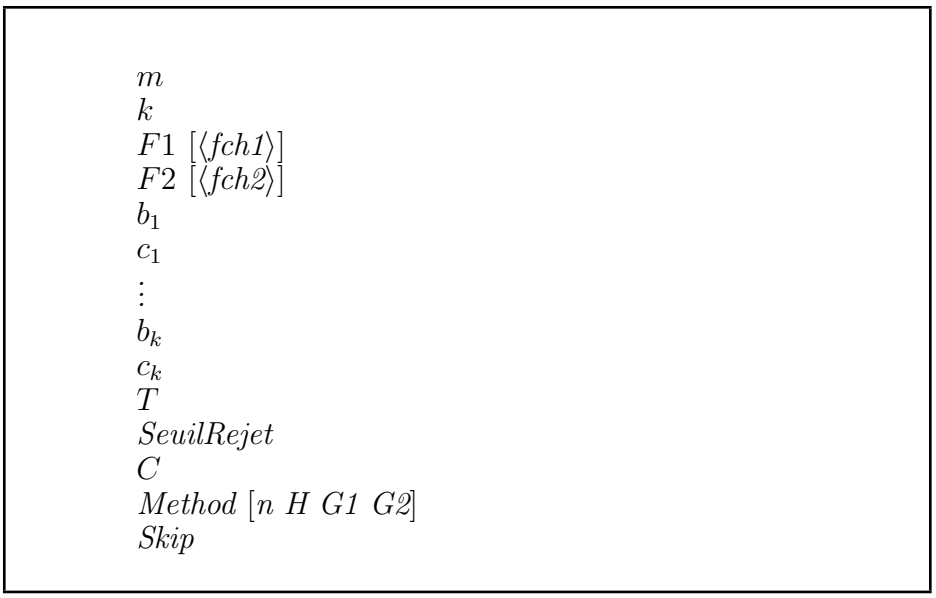

Figure 1: Format du fichier de données.

Pour lancer la recherche, faire exécuter le programme BonGCL. Entrer le nom du fichier de données sans l'extension. Les résultats seront inscrits dans un fichier ayant le même nom que le fichier de données, mais avec l'extension "`.RES`".

Les valeurs de m, b_i et c_i peuvent être écrites sous une des deux formes suivantes :

a) un grand entier s représentant la valeur directement (en base 10), suivi immédiatement du caractère de fin de ligne ⟨EOL⟩;

b) un grand entier x suivi d'un entier positif y représentable sur un mot et d'un autre grand entier z, les trois sur une même ligne et séparés par au moins un espace. Si $x \geq 0$, la valeur attribuée à s sera $x^y + z$. Si $x < 0$, la valeur attribuée à s sera $-(|x|^y + z)$. Par exemple, $(x\ y\ z) = (2\ \ 5\ \ -1)$ donnera 31, et $(x\ y\ z) = (-2\ \ 5\ \ -1)$ donnera -31 (et non pas -33).

La signification et les valeurs permises pour les champs du fichier de données sont les suivantes:

m **et** k **:**

m doit être un nombre premier strictement positif, et k, un entier tel que $1 \leq k \leq 7$. Les entiers étant représentés par des variables de type `SuperInteger` (de taille arbitraire) du logiciel SENTIERS, la limite supérieure sur m n'est fonction que de l'espace mémoire disponible.

$F1$ **et** $\langle$*fch1*$\rangle$ **:**

$F1$ indique comment il faut trouver les facteurs de $m-1$, et $\langle$*fch1*$\rangle$ est un nom de de fichier. Les valeurs permises pour le champ $F1$ sont `(Decomp, DecompWrite, Read)`. `Decomp` indique que le logiciel devra décomposer $m-1$ en facteurs lui-même (dans ce cas, le champ $\langle$*fch1*$\rangle$ est absent). Pour factoriser, le logiciel utilise la procédure `Factorize` de SENTIERS (module `SUPFACT`) avec un temps limite infini. C'est la responsabilité de l'usager de s'assurer que la factorisation prendra effectivement un temps raisonnable. `DecompWrite` indique aussi que le programme devra décomposer $m-1$, mais en plus il fera écrire les facteurs supérieurs à 1 dans le fichier $\langle$*fch1*$\rangle$, avec un seul facteur par ligne. `Read` indique que les facteurs de $m-1$ doivent être lus dans le fichier $\langle$*fch1*$\rangle$, toujours avec un seul facteur par ligne. Il n'est pas nécessaire que les facteurs soient triés, mais ceux qui se répètent doivent être sur des lignes consécutives (ils doivent se suivre). Chaque factorisation doit être complète, car le programme vérifie que le produit des facteurs du fichier est bien égal au nombre à décomposer. Si ce n'est pas le cas, il affiche un message d'erreur et s'arrête.

$F2$ **et** $\langle$*fch2*$\rangle$ **:**

Ces deux champs sont comme $F1$ et $\langle$*fch1*$\rangle$, sauf qu'ils se rapportent à la factorisation de $r = (m^k - 1)/(m-1)$. Dans ce cas-ci, il est possible que r soit premier (cela n'était pas possible dans le cas de $m-1$, car m doit être premier), auquel cas on peut mettre la valeur `Prime` pour $F2$. Les valeurs possibles pour $F2$ sont donc `(Decomp, DecompWrite, Read, Prime)`. L'usager peut utiliser les procédures `Factorize` et `IsPrime` du logiciel SENTIERS pour factoriser $(m-1)$ ou r ou vérifier la primalité de r.

$b = (b_1, \ldots, b_k)$ **et** $c = (c_1, \ldots, c_k)$ **:**

Les b_i et c_i sont des entiers tels que $-m < b_i \leq c_i < m$, pour $i = 1, \ldots, k$. Ils déterminent la zone de recherche.

T :

Ce paramètre est un entier tel que $k+1 \leq T \leq 8$. Il représente la dimension maximale pour laquelle on veut effectuer le test spectral.

SeuilRejet :

Il s'agit d'un nombre réel entre 0 et 1. Il représente la valeur minimale de M_T pour qu'un générateur soit conservé dans le rapport final.

C :

C'est le nombre maximum de générateurs à inscrire dans le rapport. C'est un entier tel que $1 \leq C \leq 99$.

Method **et** n, H, $G1$, $G2$:

Pour *Method* les valeurs permises sont `(Exhaust, Aleat)`. `Exhaust` indique que l'on va effectuer une recherche exhaustive, dans toute la région déterminée par b et c, tandis que `Aleat` indique une recherche aléatoire. Dans le cas d'une recherche exhaustive, les autres paramètres de cette ligne sont absents (ou inutilisés). Dans le cas d'une recherche aléatoire, n est le nombre de sous-régions à examiner, et H est un entier positif qui détermine la taille de chaque sous-région. Le vecteur $(h_1, \ldots, h_k)$ est déterminé par le programme comme ceci : $h_i = \min(H, c_i - b_i + 1),\ i = 1, \ldots, k$. $G1$ et $G2$ sont les germes du générateur utilisé par le logiciel pour générer les α_i. Avant de commencer la recherche, le programme effectue un appel à `SetSeed` $(G1, G2)$ (voir module `SUPALEA` de SENTIERS). Pour recommencer une recherche aléatoire dans une zone déjà étudiée et examiner des générateurs différents, il suffit de changer ces germes. On doit avoir $1 \leq G1 \leq 2147483562$ et $1 \leq G2 \leq 2147483398$.

Skip :

Les valeurs permises sont `(SkipES, NoSkipES)`. Cette variable permet, lorsqu'elle est égale à `SkipES`, de sauter l'étape "Exhaustive search in the remaining region" (voir [2], section 3) si elle risque de devenir trop coûteuse. Plus précisément, elle fait sauter par-dessus la boucle `WHILE` si un des z_j est plus grand que 1. Cette étape tend à faire grandir le temps d'exécution du test proportionnellement à $\prod_{j=1}^{T}(2z_j + 1)$, c'est pourquoi il peut être utile de la court-circuiter. Si elle est sautée pendant le test sur un générateur donné, celui-ci est rejeté puisque ses résultats ne sont plus valides. On risque ainsi de perdre quelques bons générateurs, mais le temps sauvé peut parfois permettre d'en examiner beaucoup plus au total. Pour ne jamais sauter par-dessus la boucle, choisir `NoSkipES`.

4 Exemples

La figure 2 contient `GCL1.DAT`, un exemple de fichier de données pour une recherche exhaustive de GCL d'ordre 2 avec $m = 32749$. Les valeurs de $(m-1)$ et de $r = (m^k - 1)/(m-1)$ seront décomposées par le programme. Les facteurs de $(m-1)$ seront écrits dans le fichier `GCL1_F1.FAP`, tandis que ceux de r ne seront pas conservés. La zone de recherche est limitée par $b = (1, -180)$ et $c = (180, -1)$; les 32400 vecteurs de multiplicateurs qu'elle contient seront tous examinés. On conservera les 5 meilleurs générateurs parmi ceux de période maximale qui obtiennent un résultat minimum de 0.2 jusqu'à la dimension $T = 6$. L'étape "Exhaustive search in the remaining region" sera toujours effectuée au complet. Les résultats seront écrits dans `GCL1.RES`.

```
32749
2
DecompWrite  GCL1_F1.FAP
Decomp
1
180
-180
-1
6
0.2
5
Exhaust
NoSkipES
```

Figure 2: Exemple de fichier de données : `GCL1.DAT`.

Le fichier `GCL2.DAT` (figure 3) contient les données d'une recherche aléatoire de GCL d'ordre 5 avec $m = 2^{63} - 711$. La factorisation de $(m-1)$ sera lue dans le fichier `GCL2_F1.FAP`; $(m^k - 1)/(m-1)$ est premier. Les vecteurs bornant la zone de recherche sont $b = (1, 0, 0, 0, -(2^{63} - 712))$ et $c = (2^{63} - 712, 0, 0, 0, -1)$, c'est-à-dire que les multiplicateurs a_2, a_3, a_4 sont fixés à zéro. Les 5 générateurs de période maximale ayant les meilleurs résultats au test spectral jusqu'en dimension 8 seront retenus pour le rapport final. Le seuil de rejet a été fixé à 0.0. On examinera 1000 sous-régions de dimensions $(2 \times 1 \times 1 \times 1 \times 2)$, soit 4000 générateurs au total. Les germes initiaux sont $G1 = 123456789$ et $G2 = 1234567890$. L'étape "Exhaustive search in the remaining region" sera sautée s'il y a lieu. Les résultats seront écrits dans le fichier `GCL2.RES`.

```
2  63  -711
5
Read  GCL2_F1.FAP
Prime
1
2  63  -712
0
0
0
0
0
0
-2  63  -712
-1
8
0.0
5
Aleat  1000  2  123456789  1234567890
SkipES
```

Figure 3: Exemple de fichier de données : `GCL2.DAT`.

Références

[1] Knuth, D. E. (1981). *The Art of Computer Programming, vol. 2: Seminumerical Algorithms*, second edition, Addison-Wesley.

[2] L'Ecuyer, P. et Blouin, F. (1988). "Linear Conguential Generators of Order $k > 1$", Proceedings of the 1988 Winter Simulation Conference, IEEE Press, 1988, 432–439.

[3] L'Ecuyer, P., Perron, G. et Blouin, F. (1988). SENTIERS: un logiciel Modula-2 pour l'arithmétique sur les grands entiers. Technical report DIUL-RT-8802, Université Laval.