Optimization Networks for the Generation of Block Designs

Pau Bofill
Enric Fontdecaba
Departament d'Arquitectura de Computadors—UPC
Carme Torras
Institut de Cibernètica—(CSIC-UPC)

The generation of Block Designs—matrix configurations satisfying certain constraints—constitutes a good benchmark for the evaluation of optimization strategies. Three such strategies, built on 4th-order neural networks, have been implemented on a CM-200 multiprocessor and compared on the 25 smallest design problems. Simulated Annealing (SA) beats Down-Hill search (DH) both in the ratio of success and the expected cost of a solution, with 23 solved problems against 12 for DH. The third strategy is a new deterministic cooperative search, which we call Parallel Mean Search (PMS), that searches for progressively deeper low-energy regions by sampling the cost function in *parallel*, and moves in the direction of decreasing *mean* energy, like a big sliding "ball". The radius of the ball is progressively reduced until convergence to DH. PMS solved 21 problems and its results were compared with a clustered reinterpretation of DH. On the problems solved by both, PMS showed more often a higher ratio of success, at the expense of a higher expected cost. SA was superior to the other two, in both senses. Being still non-mature, PMS results were nevertheless promising.

Block Designs, Down-hill Search, Simulated Annealing, Parallel Mean Search, Performance Comparison.

[0] Correspondence and requests for reprints should be sent to Pau Bofill, Departament d'Arquitectura de Computadors-UPC. Campus Nord, Mòdul D6. c/Gran Capità s/n. 08071 Barcelona. Email:pau@ac.upc.es.

1 INTRODUCTION

The generation of Block Designs is an NP-complete problem with difficult to find, easy to check solutions. Designs with different parameters and sizes behave like entirely different problems, rather than like different instances of the same problem. Some are easy to solve. Others are still unsolved. Design generation can be formulated as the optimization of a fourth-order cost function. Despite the optimum cost being known beforehand, combinatorial explosion arises even for small designs. All these facts make Block Design a very good benchmark for the evaluation of optimization strategies.

A *Balanced Incomplete Block Design* (BIBD) [7] is an arrangement of *elements* of a set in several subsets called *blocks* in such a way that every element belongs to the same number of blocks, every block has the same number of elements, and every pair of elements co-occur in the same number of blocks. The *incidence matrix* A of a BIBD with v elements and b blocks is a $v \times b$-binary matrix with exactly r ones per row and exactly k ones per column, such that the dot product (or correlation) of every pair of rows is exactly λ. The constants v,b,r,k,λ are called the *parameters* of the design, and the resulting configuration is called an (v,b,r,k,λ)-BIBD.

The parameters of a block design are said to be *admissible* when the following two conditions are met:

$$vr = bk, \tag{1}$$

$$r(k-1) = \lambda(v-1). \tag{2}$$

The admissibility of the parameters is a necessary but not sufficient condition for the *existence* of a BIBD. For a given set of admissible parameters, the number of existing non-isomorphic BIBDs is not known in the general case. For some parameter sets it can be proved that no designs exist, for some others the number of existing non-isomorphic designs is known (or at least a lower bound on it), and for the rest neither existence nor in-existence has yet been proved [9].

For some families of designs there exist constructive methods of two kinds, direct and recursive [7], and different algorithmic techniques have also been used [6, 3]. Algorithmic generation of BIBDs is NP-complete [3].

Like many combinatorial configurations, BIBDs have their origins in the field of experimental design [4] (arranging experiments in such a way that the statistical analysis of their results is well defined).

2 AN OPTIMIZING FOURTH-ORDER NETWORK FOR THE GENERATION OF BLOCK DESIGNS

Let us define a cost function

$$\begin{aligned} F([x_{pq}]) &= Q - \theta o = \\ &= \sum_{i=1}^{v} \sum_{k>i}^{v} \sum_{j=1}^{b} \sum_{l>k}^{b} x_{ij} x_{il} x_{kj} x_{kl} - \theta \sum_{i=1}^{v} \sum_{j=1}^{b} x_{ij} \end{aligned} \tag{3}$$

over the set $\mathcal{A}$ of all possible $v \times b$-binary matrices or *configurations* of the kind $A \equiv [x_{ij}]$, where $x_{ij} \in \{0,1\}$ represents the incidence of element i in block j, Q is the number of active *quadruples*, and o is the total number of ones.

The cost increment relative to x_{ij} is

$$\begin{array}{rcl} \Delta F_{ij} & = & F|_{x_{ij}=1} - F|_{x_{ij}=0} = \\ & = & q_{ij} - \theta \\ & = & \sum_{k=1,k\neq i}^{v} \sum_{l=1,l\neq j}^{b} x_{il} x_{kj} x_{kl} - \theta, \end{array} \tag{4}$$

where q_{ij} is the number of *potential* quadruples to which x_{ij} belongs.

In the following we state that, if there exists a (v,b,r,k,λ)-BIBD with incidence matrix A^*, a value for θ can be found that ensures that $F(A^*)$ is a *global minimum* of F over $\mathcal{A}$. Let

$$\begin{array}{rl} \theta^1 = & (k-1)(\lambda-1) \\ \theta^0 = & k\lambda. \end{array}$$

Proposition 2.1 *There exists a (v,b,r,k,λ)-BIBD with incidence matrix A^** if and only if

$$q_{ij} = \left\{ \begin{array}{ll} \theta^1 & \forall (i,j) : x_{ij} = 1 \\ \theta^0 & \forall (i,j) : x_{ij} = 0. \end{array} \right.$$

Proposition 2.2 If $\theta = \frac{1}{2}(\theta^1 + \theta^0)$ then

i) *If there exists a (v,b,r,k,λ)-BIBD with incidence matrix A^*, then A^* is a global minimum of F over the set $\mathcal{A}$ of all $v \times b$-configurations and $F(A^*) = Q^* - \theta u$, where $Q^* = \binom{v}{2}\binom{\lambda}{2}$ and $u = vr = bk$.*

ii) *If there exists a $v \times b$-configuration A^* such that $F(A^*) = Q^* - \theta u$, then A^* is the incidence matrix of a (v,b,r,k,λ)-BIBD.*

The prove can be found in [2].

The mapping of the cost function in (3) onto a neural network is straightforward if we use fourth-order connections to represent quadruples.

We define a network with $v \times b$ binary units arranged in a two-dimensional array, in direct correspondence with the $[x_{ij}]$ incidence matrix, with connections defined as follows:

- *Quadruples*, fourth order symmetric constraints relating square-wise arranged units: $\{(x_{ij}, x_{il}, x_{kj}, x_{kl}) | k \neq i, l \neq j\}$.
- *Biases*, the usual first order constraints: $\{x_{ij}\}$.

Then, if we define the *energy* of the network in the usual way (the sum of the weights of all active connections), and we set all quadruple weights to 1 and all bias weights to $-\theta$, then (3) is the energy of a particular state of the network, and (4) is the *local field* to unit (i,j) in that state. Notice that, because of the definition of the cost increment, the actual energy increment is given by

$$\Delta E_{ij} = F|_{\overline{x}_{ij}} - F|_{x_{ij}} = (1 - 2x_{ij})\Delta F_{ij}. \tag{5}$$

3 PARALLEL MEAN SEARCH

The simplest search strategy for optimization networks is DH [8], that updates a unit whenever the overall energy of the network decreases. In order to avoid the local minima of DH, the assumption behind SA [1] is that global minima lie in the regions of search space where the average energy is low. With the aid of thermal noise, energy barriers disappear because uphill moves are equally likely than downhill moves. Reducing the computational temperature, then, leads the network into progressively deeper regions of average energy.

Parallel Mean Search exploits the same idea in a different way. Instead of sampling average energies with a single stochastic instance of the network, PMS uses a set of network instances, a *cluster*, and explores search space in parallel, in a deterministic way. Each member of the cluster is initialized at a different initial state and the average energy is defined as the *mean* energy of the members of the cluster. Then, all members move in strict formation in the direction of decreasing mean energy, like a big sliding ball. For every member m, then, the PMS decision rule is,

$$x_{ij}^m \leftarrow \overline{x}_{ij}^m, \quad \text{when} \quad \sum_{n=1}^{S} \Delta E_{ij}^n < 0, \tag{6}$$

with the same (i,j) for all cluster members, and S the size of the cluster.

The *radius* R of the cluster is defined as the maximum Hamming distance between any cluster member and a particular one, called the *reference*. A schedule is then devised to reduce the radius from an initial value R_0 to a final value R_f, gradually reducing the size of the region explored (the size of the ball). Thus, at each radius step, "concentric" regions of search space are explored, in accordance to the original assumption. When R_f is met, all members are let free to relax to their nearest DH minimum.

The cluster is defined to *succeed* (in finding a solution) if at least one of the local minima is an optimum.

The general idea outlined above leaves room to many different alternatives, relating the structure of the clusters, the radius update schedule and the parameter tuning. The next section shows experimental results for only a few of them.

Table 1: Problem number, design parameters, design size vb and number A of non isomorphic solutions, for all admissible parameter sets with $vb \leq 300$.

#	v	b	r	k	λ	vb	A	#	v	b	r	k	λ	vb	A
1	7	7	3	3	1	49	1	14	7	28	12	3	4	196	35
2	6	10	5	3	2	60	1	15	9	24	8	3	2	216	36
3	7	14	6	3	2	98	4	16	8	28	14	4	6	224	2224
4	9	12	4	3	1	108	1	17	15	15	7	7	3	225	5
5	8	14	7	4	3	112	4	18	6	40	20	3	8	240	13
6	6	20	10	3	4	120	4	19	11	22	10	5	4	242	4393
7	11	11	5	5	2	121	1	20	7	35	15	3	5	245	108
8	7	21	9	3	3	147	10	21	16	16	6	6	2	256	3
9	10	15	6	4	2	150	3	22	12	22	11	6	5	264	601
10	9	18	8	4	3	162	11	23	7	42	18	3	6	294	417
11	13	13	4	4	1	169	1	24	10	30	12	4	4	300	998
12	10	18	9	5	4	180	21	25	10	30	9	3	2	300	960
13	6	30	15	3	6	180	6								

PMS was specifically tailored for SIMD multiprocessor systems. With each processor running a different cluster member, cooperation would be achieved with a high degree of parallelism at the expense of a low communication cost.

4 EXPERIMENTS AND RESULTS

DH, SA and PMS where implemented on a CM-200 Connection Machine [11], with $N = 2048$ bit serial processors, and run on all BIBD admissible parameter sets with $v \times b \leq 300$. Table 1 lists the problem number, the parameters of the design, its size $(v \times b)$ and the number A of non-isomorphic solutions, for a total of 25 problems (from [9]).

The three search strategies were compared in terms of the ratio of success ρ (i.e., the ratio of global to local minima) and in terms of the expected number of iterations to the first solution $\overline{z}$ (the complexity of an iteration is, roughly, $(v \times b)^2$, that is, $v \times b$ the computational cost of equation (4)). Since cluster members are not independent from each other, PMS had to be compared with a clustered reinterpretation of DH and SA results. A detailed description of these performance measures, detailed results and confidence margins, can be found in [2].

A total of 2048 descents of the DH strategy were performed on each of the 25 BIBD problems. In 13 of them (mainly the larger ones) DH failed to find any solution at all. Yet, where solutions were found, there was no correlation between the ratio of success and the problem size. Some large problems proved to be relatively easy to solve. The average descent cost showed only a slow and irregular tendency to increase with problem size (in the range from 3 to 7 iterations), and,

Table 2: DH and SA compared in terms of the ratio of success ρ, and in terms of the expected number of iterations to the first solution $\overline{z}$. Stars show best marks.

	$100 \times \rho$		$\overline{z}$	
#	DH	SA	DH	SA
1	25.39	$*$ 97.56	13	$*$ 10
2	50.20	$*$ 91.36	$*$ 7	13
3	28.81	$*$ 76.32	$*$ 15	25
4	0.44	$*$ 16.26	1049	$*$ 196
5	3.03	$*$ 13.48	$*$ 142	249
6		$*$ 10.79		$*$ 313
7	2.59	$*$ 21.34	200	$*$ 150
8		$*$ 18.75		$*$ 169
9	0.49	$*$ 6.01	1054	$*$ 577
10	0.29	$*$ 1.71	$*$ 1652	2088
11	7.37	$*$ 46.19	77	$*$ 58
12		$*$ 0.05		$*$ 73713
13		$*$ 1.76		$*$ 2026
14		$*$ 1.22		$*$ 2932
15	2.98	$*$ 31.54	193	$*$ 94
16		$*$ 0.10		$*$ 36845
17		$*$ 0.10		$*$ 36842
18		$*$ 0.63		$*$ 5653
19				
20		$*$ 0.49		$*$ 7356
21	0.05	$*$ 1.56	12976	$*$ 2286
22				
23		$*$ 0.15		$*$ 24559
24		$*$ 0.05		$*$ 73705
25	1.07	$*$ 14.26	577	$*$ 234

therefore, the main contribution to $\overline{z}$ was due to the ratio of success.

Unlike DH, the ratio of success in SA depends on the temperature schedule. But so does the execution cost. Thus, a good parameter tuning was necessary. The usual exponential schedule was used, $T_{n+1} = kT_n$, and a set of preliminary experiments was performed to select the following parameters: when no solution was found, the final temperature T_f was set to the highest experimental value that, for *each* BIBD problem, would not drop the ratio of success; the initial temperature was set to $T_0 = 9T_f$; and the decay constant was selected to be $k = 0.939$. That way, the maximum number of iterations would be 35 [2]. For the decision rule, the Metropolis algorithm was used [10].

Like before, 2048 SA descents were run for each of the 25 problems. Table 2 compares SA with DH in terms both of the ratio of success ρ, and the expected number of iterations to the first solution $\overline{z}$. Stars show best marks. Only 2 problems remained unsolved for SA. The improvement in ρ was amazing. And results with SA showed a noticeble dependency with problem size, suggesting a better scalability. In spite of a higher descent cost, SA still showed lower $\overline{z}$ than DH in 8 out the 12 cases that DH could solve.

For the PMS strategy, parameter tuning was another matter. On the one hand, very little information was available to guide our intuition (only a set of preliminary experiments reported in [5]). And, on the other hand, statistically significant experimental results were much more costly. Thus, cluster size was arbitrarily set to $S = 32$, and 7 different parameter settings were selected, PMS1 to PMS7, that used different combinations for cluster structure and radius schedule (see [2] for details).

PMS1 to PMS7 were run 128 times on each of the 25 problems (128 *cluster* descents each time). And, for each problem, the setting that showed best results was selected. Table 3 shows PMS results as compared with a clustered reinterpretation of DH (DH runs were clustered together by 32's and a cluster was redefined to succeed if at least one of its members was an optimum). PMS was clearly superior in the number of solved problems (21 against 12). And, for the 12 problems that both algorithms could solve, PMS scored 3 to 1 significantly better success ratios. Yet, in terms of the expected cost, PMS could only once outperform the low descent cost of DH.

In view of the above results, the comparison between PMS and a clustered reinterpretation of SA showed no surprises, and the corresponding table is omitted here [2]. In terms of the number of solved problems, SA beated PMS in two ocasions. Yet, neither of them was significant because of different sample sizes (2048 for SA against 128 for PMS). Nevertheless, SA proved always superior, in terms both of ρ and of $\overline{z}$.

From the point of view of problems, there were remarkable differences independent of problem size. We found that the largest problem was one of the easiest to solve (Prbl. #25). Whereas a medium size problem, like #12, was one of the most difficult to solve. In all, only two problems (#22 and #24) remained unsolved.

5 CONCLUSIONS AND FURTHER WORK

This paper presented a comparison between three optimization algorithms: Down-Hill search (DH), Simulated Annealing (SA), and a new cooperative search strategy called Parallel Mean Search (PMS). The three algorithms were implemented on a Connection Machine with 2048 multiprocessors. The benchmark were 25 instances of the generation of BIBDs, namely all admissible BIBD parameter sets in [9] with size $v \times b$ less than 300. A cost function for the generation of BIBDs was described, which mapped straightforwardly onto a network with square-wise 4th-order connections and 1st-order biases.

For a given algorithm and problem, the main performance measures were the

Table 3: Clustered-DH and PMS compared in terms of the ratio of success ρ, and in terms of the expected number of cluster iterations to the first solution $\overline{z}$. Stars show best marks.

	$100 \times \rho$		$\overline{z}$	
#	clustered-DH	PMS	clustered-DH	PMS
1	100.00	100.00	$*$ 104	399
2	100.00	100.00	$*$ 113	396
3	100.00	100.00	$*$ 141	430
4	14.06	$*$ 56.25	1049	$*$ 763
5	59.38	57.81	$*$ 232	762
6		$*$ 11.72		$*$ 3116
7	51.56	58.59	$*$ 321	788
8		$*$ 8.59		$*$ 13589
9	15.62	19.53	$*$ 1054	2321
10	7.81	12.50	$*$ 1983	3570
11	90.62	100.00	$*$ 201	444
12				
13		$*$ 14.06		$*$ 8160
14		$*$ 6.25		$*$ 18536
15	$*$ 64.06	$*$ 94.53	$*$ 288	473
16		$*$ 0.39		$* < \infty$
17		$*$ 0.16		$* < \infty$
18		$*$ 6.25		$*$ 18577
19				
20		$*$ 5.47		$*$ 21393
21	1.56	0.78	$*$ 12976	58823
22				
23		$*$ 0.78		$*$ 149793
24				
25	31.25	$*$ 56.25	634	777

ratio of success (the ratio of global to local minima) and the expected number of iterations to the first solution. In order to compare PMS, a parallel strategy, with DH and SA, clustered equivalent performance measures were devised for the latter two. While, for SA, critical temperatures and the schedule of temperature decays were well tuned, the blind series of parameter settings tried for PMS didn't turn out to be satisfactory enough.

A large variability showed up on the part of the problems, and only SA performance seemed to degrade regularly with problem size, thus scaling up quite well. DH failed on 13 of the 25 problems, PMS on 4, and SA on 2. The 2 extra problems solved by SA were not actually significant, due to the different sample sizes used. In all, 2 problems remained unsolved. When two strategies succeeded to solve a given problem, SA outperformed the other two, both in the ratio of success and in the expected cost. PMS proved to be slightly more efficient than a clustered reinterpretation of DH results in terms of success ratio, at the expense of a higher expected cost.

Being a non mature strategy, results with PMS have been interesting, especially where success ratios are concerned. Further work includes a finer parameter tuning, in order to find a critical radius and a cluster size and structure that might yield better results for each particular problem. Many alternatives are left to explore, such as different radius schedules (e.g., exponential decrease might be considered), cluster size scheduling (either leaving out bad members or gradually decomposing clusters into progressively smaller ones), SA applied to PMS, and so on. A careful experimental design would help to cope with so many possibilities. Like SA, PMS behaviour is based on the average energy landscape. Yet a deeper "physical" interpretation is necessary to provide better insight.

References

[1] E.H.L. Aarts, and J.H.M Korst. *Simulated Annealing and Boltzmann Machines*. Wiley Interscience, 1988.

[2] P. Bofill, E.Fontdecaba, and C. Torras. Optimization Networks for the Generation of Block Designs. *Technical Report* UPC-DAC-1995-41. Available via ftp at gaudi.ac.upc.es/pub/reports/DAC/1995/UPC-DAC-1995-41.ps.Z.

[3] D.G. Corneil, and R.A. Mathon. Algorithmic Techniques for the Generation and Analysis of Strongly Regular Graphs and Other Combinatorial Configurations. In *Ann. of Discrete Mathematics*, North Holland Publishing Company, Vol. 2, pp. 1-32, 1978.

[4] R.A. Fisher. An Examination of the Different Possible Solutions of a Problem in Incomplete Blocks. In *Ann. Eugen.*, Vol 10, p 52-57, 1940.

[5] F. Fontdecaba. *Algorismes per a la cerca de BIBD's.* Projecte Final de Carrera, E.T.S.E. Telecomunicació de Barcelona. Supervised by Pau Bofill. Setembre 1994.

[6] P.B. Gibbons, R.A. Mathon, and D.G. Corneli. Computing Techniques for the Construction and Analysis of Block Designs. In *Utilitas Mathematica*, Vol 11, pp 161-192, 1977.

[7] M. Hall. *Combinatorial Theory.* John Wiley & Sons, Second Edition 1986.

[8] J.J. Hopfield. Neural Networks and Physical Systems with Emergent Collective Computational Abilities. In *Proc. Nat. Academ. Sciences USA*, Vol 79, 2554-2558, 1982.

[9] R. Mathon, and A. Rosa. Tables of parameters of BIBD with r$\leq$41 including existence, enumeration and resolvability results: an update. In *Ars Combinatoria*, Vol 30, December, Winnipeg, Canada, 1990.

[10] N. Metropolis, A.W. Rosenbluth, M.N. Rosenbluth, A.H. Teller, and E. Teller. Equation of State Calculations for Fast Computing Machines. In *Journal of Chemical Physics*, Vol 21, pp 1087-1092. 1953.

[11] Thinking Machines Corporation. *CM-200 Technical Summary*. June 1991.