

Dynamic Weighting Monte Carlo for Constrained Floorplan Designs in Mixed Signal Application

Jason Cong*, Tianming Kong*, Faming Liang[†], Jun S. Liu[‡], Wing Hung Wong[†], Dongmin Xu*

*Department of Computer Science
University of California
Los Angeles, CA 90095
{cong,kongtm,dongmin}@cs.ucla.edu

[†] Department of Statistics
University of California
Los Angeles, CA 90095
{fmliang,whwong}@stat.ucla.edu

[‡] Department of Statistics
Stanford University
Stanford, CA 94305
jliu@stat.stanford.edu

Abstract

Simulated annealing has been one of the most popular stochastic optimization methods used in the VLSI CAD field in the past two decades. Recently, a new Monte Carlo and optimization method, named dynamic weighting Monte Carlo [WL97], has been introduced and successfully applied to the traveling salesman problem, neural network training [WL97], and spin-glasses simulation [LW99]. In this paper, we have successfully applied dynamic weighting Monte Carlo algorithm to the constrained floorplan design with consideration of both area and wirelength minimization. Our application scenario is the constrained floorplan design for mixed signal MCMs, where we need to place all the analog modules together in groups so that they can share common power and ground planes, which are separate from those used by the digital modules. Our experiments indicate that the dynamic weighting Monte Carlo algorithm is very effective for constrained floorplan optimization. It outperforms the simulated annealing for a real mixed signal MCM design by 19.5% in wirelength, with slight area improvement. This is the first work adopting the dynamic weighting Monte Carlo optimization method for solving VLSI CAD problems. We believe that this method has applications to many other VLSI CAD optimization problems.

I. INTRODUCTION

Simulated annealing (SA) [KG83] has been one of the most popular stochastic optimization methods used in the VLSI CAD field in the past two decades. However, the efficiency of simulated annealing depends much on the energy landscape. If the global minimum solution has a small basin of attraction and is well separated by many

local minima with high energy barriers, simulated annealing tends to be trapped in a local minimum solution. The reason is that in practice, to assure reasonable runtime, the temperature cooling schedule used in the simulated annealing process is exponential cooling schedule [SS94], which is much faster than the logarithmic cooling schedule required by theory [GG84] for achieving optimality.

In early 1990's, *simulated tempering* (ST) [MP92, GT95] was introduced to overcome the drawbacks of simulated annealing by taking temperature as an additional random variable during the optimization process. Although simulated tempering can help the system jump out of local minima in the appearance of high energy barriers, it sometimes suffers from the serious problem that the energy distribution of two adjacent temperature levels cannot mix well, which means that the temperature levels have to be very closely spaced in order to preserve smooth temperature transition. As a result, too many temperature levels have to be used to explore a large temperature range in order to sample good solutions, which will result in long computation time. An extension of simulated tempering, called *dynamic weighting Monte Carlo* (DWMC) [WL97], in which a weighting variable is employed to help the system jump between adjacent temperature levels, has been successfully applied to the traveling salesman problem, neural network training [WL97], and spin-glasses simulation [KR94, LW99]. For two benchmarks (*pcb442* and *att532*) [Re95] of traveling salesman problem (with optimal values of 50778 and 27686), dynamic weighting algorithm has obtained the solutions which are over the optimal values by 0.097% and 0.114%, respectively; while the solutions of simulated annealing are over the optimal values by 1.264% and 2.644% [Li97], respectively.

Since dynamic weighting Monte Carlo is a general optimization method, it can be used to solve many optimization problems. In this paper, we have successfully applied dynamic weighting Monte Carlo approach to the optimization of slicing floorplan for mixed signal MCM designs. To deal with this problem, we have to place all the

*The work of Jason Cong, Tianming Kong, Dongmin Xu is partially supported by DARPA/ETO under Contract DAAL01-96-K-3600, NSF Young Investigator Award MIP9357582.

[†]W.H. Wong's work is supported by NSF grant DMS-9703918.

[‡]J.S. Liu's work is supported by NSF grant DMS-9803649.

analog modules together in groups so that they can share common power and ground planes, which are separate from those used by the digital modules. Also, the clustering of analog modules makes it easy to provide shielding for these analog modules for signal integrity. Since the constrained floorplan problem usually introduces many dramatic configuration changes, it results in more deep local minima to trap simulated annealing algorithm. Our experiments indicate that the dynamic weighting Monte Carlo algorithm is especially effective in the constrained floorplan designs. For a real mixed signal MCM design of a high speed wireless modem, dynamic weighting Monte Carlo based approach outperforms the simulated annealing based approach by 19.5% in wirelength, while gets slight area improvement.

The rest of this paper is organized as follows: Section II reviews the formulation of slicing floorplan approach. Section III describes the dynamic weighting Monte Carlo algorithm. Section IV discusses our algorithm for mixed signal MCM designs. Section V shows the experimental results. Section VI gives the conclusion.

II. FORMULATION OF SLICING FLOORPLAN DESIGNS

There are two approaches for the floorplan problem: slicing versus non-slicing. Both slicing floorplan [WL89, YT96] and non-slicing floorplan approaches [PL93, MF95, NF96] have been investigated extensively. Compared with non-slicing floorplan approaches, the slicing floorplan approach is efficient, easy to implement and produces even better solutions [YT96, YW97]. In [YT96], the slicing approach outperforms the non-slicing approach [MF95] for an MCNC circuit *ami49* by 4.8% and 23.0% in area and wirelength, respectively. In this work, we only consider slicing floorplan as well.

In this paper, we assume basic modules are all rectangular. For a given set of modules $M = \{m_1, m_2, \dots, m_n\}$, each module m_i can be represented by a triplet (A_i, l_i, u_i) , where A_i is the area of module m_i ($1 < i < n$), l_i and u_i specify lower and upper bounds of module i 's aspect ratio. If $l_i = u_i$, then module m_i is said to be rigid, otherwise, it is flexible. A floorplan for the given n modules consists of a bounding rectangle, partitioned by some horizontal and vertical line segments into n non-overlapping rectangular regions, denoted by r_1, r_2, \dots, r_n . Each region r_i must be large enough to accommodate its module m_i . For the mixed signal MCM designs, the analog modules should be put together in groups. The definition of the problem can be stated as follows:

Definition 1: Given a set of modules M , a subset analog modules $S \subseteq M$ and an integer k , compute a slicing floorplan such that all modules in S are clustered in no more than k rectangular regions (called "analog regions") and the weighted sum of area and wirelength of the floorplan is minimized.

Note that the digital modules in $M - S$ should not appear in the analog regions. Fig. 1 illustrates a simple mixed signal MCM design. The dark blocks represents analog modules, while these big light blocks are digital modules. Usually, a typical mixed signal MCM design has a lot of small analog modules and a few large digital modules.



Fig. 1. Illustration of a simple mixed signal MCM design

Polish expression representation was introduced for the slicing floorplan in [WL89]. A slicing floorplan is a floorplan which can be obtained by recursively partitioning a rectangular region into two parts either by a vertical line or a horizontal line. The resulting area dissection corresponds to a slicing tree, in which each leaf represents a region r_i ($1 < i < n$) and each internal node represents a cut line. Let horizontal and vertical cut be denoted by the operator $+$ and $*$, respectively; and the modules be denoted by operands. There exists a one-to-one mapping from the set of slicing trees to the set of normalized Polish expressions. To explore different floorplan configurations using Polish expression approach, three type of moves, $M1$, $M2$, $M3$, were defined in [WL89]. Operation $M1$ swaps two adjacent operands; Operation $M2$ interchanges the operators $*$ and $+$ for a chain of nonzero length of adjacent operators; Operation $M3$ swaps two adjacent operand and operator.

Suppose that a floorplan configuration is represented by \mathcal{F} , and total area and wirelength of \mathcal{F} are denoted by $A(\mathcal{F})$ and $W(\mathcal{F})$. The cost function given in [WL89] is as follows:

$$f(\mathcal{F}) = A(\mathcal{F}) + \lambda W(\mathcal{F}) \quad (1)$$

The simulated annealing [KG83] algorithm was used to explore different configurations using cost function defined by equation (1).

III. DYNAMIC WEIGHTING MONTE CARLO ALGORITHM

There is a deep connection between global optimization and Monte Carlo simulation. In optimization, our goal is to search for a configuration x that minimizes some cost function $f(x)$. In Monte Carlo, we attempt to sample the configuration x according to a Boltzmann probability density

$$\pi_i = \alpha_i e^{-\frac{f(x)}{t_i}} \quad (2)$$

where at a fixed temperature $t = t_i$, α_i is the normalizing constant. If t_i is small, then with high probability the sampled configuration will have a cost close to the globally minimal cost value. This is the basis of simulated annealing. It has been observed that for practical annealing scheme (for example, geometric decrease of temperature), SA tends to be trapped in a local minimum. To overcome this difficulty, DWMC [WL97] employs advanced Monte Carlo technique to sample configurations at a fixed ladder of temperatures $t_1 > t_2 > \dots > t_m$ in order to sample low energy configurations. Similar to SA, it is an iterative process that produces a sequence of configurations. At step j in the iteration, there is a temperature level $t(j)$ and the corresponding configuration $x(j)$ can be thought of as being sampled from the Boltzmann distribution (2) with temperature $t(j)$ which takes value in $\{t_1, t_2, \dots, t_m\}$. However, unlike SA, the temperature $t(j)$ is not a deterministic and monotonically decreasing sequence but rather it is treated as a part of the system to be updated together with the configuration $x(j)$. Specifically, the algorithm alternates between attempts to change the configuration x and the temperature t . The moves involving the change of configuration x with the temperature level fixed are made in exactly the same way as in SA, i.e. a proposal to move to a new configuration is accepted or rejected stochastically based on the energy difference between the new and the old configurations, relative to the temperature. The moves involving temperature change with the configuration fixed are more complicated and will be discussed in more detail below. The important property of these moves is that they ensure the generation of a sequence of configurations, each with an attached temperature and a weighting factor, so that expectations with respect to the Boltzmann distribution at any temperature level t_k can be estimated by a weighted average of the values sampled at that temperature level. Thus, in this scheme it is possible for the temperature to decrease to a low value t_i to sample low energy configurations and then increase back to a high value t_j where the configuration can undergo large changes and to escape from local traps. However the temperature changes are subject to strict stochastic transition rules in order to ensure that the Boltzmann distribution is sampled correctly at all temperature levels, thus maintaining the crucial connection between Monte Carlo simulation and global optimization.

We now turn to the detailed specification of the transition rules for temperature changes. Suppose the current temperature is $t = t_i$ and the current configuration is x with a weighting factor w . First we propose to move the temperature either up or down one level with equal probability, except at the two extreme temperature level where the proposal can only be going to the adjacent level. We then compute a quantity

$$r = c \frac{\alpha_j}{\alpha_i} e^{-f(x)(\frac{1}{t_j} - \frac{1}{t_i})} \quad (3)$$

where t_j is the proposed temperature level and $c = 1/2$ when t_i is an extreme temperature level and $c = 1$ otherwise. This proposal is accepted with probability $\min\{1, \frac{wr}{wr+1}\}$. If the proposal is accepted, the temperature is moved from t_i to t_j and the weight is changed from w to $wr + 1$. If the proposal is rejected, then the temperature remains at t_i but the weight is changed from w to $w(wr + 1)$.

For this method to work it is important to use an appropriate temperature ladder. In general, the high temperature t_1 is chosen so that large movements of the configuration is possible at that temperature, and the low temperature t_m is chosen so that configurations with cost higher than the global minimum by a large magnitude (large relative to t_m) will be unlikely to be sampled from the Boltzmann distribution. We note that with the weights set to be identically 1, the algorithm reduces to another simulation scheme known as simulated tempering (ST) [MP92]. With ST often many intermediate temperature levels are needed before the proposals for temperature changes (in either directions) are accepted with reasonably high probability. With help of the weighting factor, DWMC typically works well even with a modest number of levels (say, 10-20). Since computational complexity grows quadratically with the number of levels, this is a very useful feature of DWMC. The values of the intermediate levels and the adjustable constants are usually obtained from a short pilot run using a learning algorithm. Further details on the theory of DWMC and numerical results on various examples from machine learning, combinatorial optimization and statistical physics can be found in [WL97, LW99].

IV. IMPLEMENTATION OF DWMC FOR FLOORPLAN DESIGN

We modified the Polish expression representation approach [WL89] and applied the dynamic weighting Monte Carlo algorithm for the slicing floorplan optimization. Two new operations are defined. Operation $M1'$ swaps *any* two randomly selected operands; Operation $M3'$ swaps *any* two randomly selected operand and operator, if the resulting Polish expression is still normalized. The original $M2$ operation stays the same. By changing operation $M1$ and $M3$ to $M1'$ and $M3'$, we can make some global configuration change and speed up the algorithm.

For the constrained floorplan of mixed signal MCM designs, suppose there is a digital or analog module $m_i \in M$, the corresponding operand of module m_i in the Polish expression is denoted by o_i . If module m_i is an analog module (i.e. $m_i \in S$), the analog region it belongs to is denoted by $R(m_i)$. We define all the operands of analog modules in region $R(m_i)$ to be an analog operand group, denoted by $g(o_i)$. Because we want to place all analog modules in $R(m_i)$ together, we should let the operands in group $g(o_i)$ form a subtree in the whole slicing tree. This subtree can be represented by a consecutive subsequence of operands and operators, denoted by $s(g(o_i))$, in the entire Polish expression. If the original entire Polish expression is normalized, this subsequence of Polish expression is still normalized. The corresponding rectangle enclosing all modules of this subtree can be viewed as a big complex module.

In order to handle constrained floorplan design, operation $M1'$, $M2$, and $M3'$ have to be modified. Operation $M1'$ involves swapping two operands. Suppose the two randomly selected operands are o_i and o_j . the variation of $M1'$ operation for the constrained floorplan design can be described as follows:

- If both operands are in the same analog group or not in any groups, perform $M1'$ operation as in the unconstrained case.
- If both operands are in the different groups, perform $M1'$ operation for subsequence $s(g(o_i))$ and sequence $s(g(o_j))$.
- If one operand o_i (or o_j) is in group $g(o_i)$ (or $g(o_j)$), and another one is not in any analog groups, do $M1'$ for subsequence $s(g(o_i))$ (or $s(g(o_j))$) and operand o_j (or o_i).

Operation $M3'$ involves swapping an operand and an operator, which can be treated similarly as in operation $M1'$.

For a floorplan \mathcal{F} , the commonly used cost function is given in equation (1). However, two terms $A(\mathcal{F})$ and $W(\mathcal{F})$ in the cost function (1) may be of very different scale. One often has to adjust coefficient λ for each individual floorplan instance to obtain result with both small area and short wirelength. We choose to use the following normalized cost function in this paper.

$$f(\mathcal{F}) = \gamma \frac{A(\mathcal{F})}{A_{ref}} + (1 - \gamma) \frac{W(\mathcal{F})}{W_{ref}}; \quad (0 \leq \gamma \leq 1) \quad (4)$$

where γ is a constant between 0 and 1; A_{ref} and W_{ref} are pre-calculated by a very fast simulated annealing run. If the area and wirelength are considered equally important, we can set γ to be 0.5. In our current implementation, routing area is not considered.

TABLE I
TEST CIRCUITS

circuit	#modules	# nets	# pads
apte	9	97	73
xerox	10	203	2
hp	11	83	45
ami33	33	123	43
ami49	49	408	22
playout	62	2506	192

TABLE II
RESULTS BEFORE AND AFTER SIMULATED ANNEALING

circuit	Before SA		After SA	
	area(mm ²)	WL(μm)	area(mm ²)	WL(μm)
apte	48.50	226614	48.50	226412
xerox	21.21	405829	20.43	383909
hp	10.00	128986	9.58	118472
ami33	1.40	53964	1.29	45923
ami49	41.20	896539	42.23	670845
playout	102.81	5684780	97.35	4575894

V. EXPERIMENTAL RESULTS

We have implemented simulated annealing and DWMC algorithms for both unconstrained and constrained mixed signal floorplan designs. In our implementation, the coefficient γ in the cost function (4) is set to be 0.5. For DWMC algorithm, the number of sampled solutions is set to be 5. For unconstrained floorplan designs, we tested six MCNC benchmarks, which are listed in Table I. For constrained floorplan design, we tested a real high speed wireless modem in mixed signal MCM design, which contains 133 modules and 245 nets.

We designed our experiment as follows: we ran simulated annealing algorithm 25 times and reported the best result. For DWMC algorithm, we ran it 5 times, each time we pick the best five uncorrelated solutions and subject each to a fast simulated annealing at low temperature. This allows us to have roughly the same total run time between simulated annealing and DWMC algorithms. We show floorplan results before and after applying simulated annealing algorithm for unconstrained MCNC floorplan design in Table II. It is clear from the data that DWMC provides good sampling points and the final fast simulated annealing step can further improve the results.

For unconstrained floorplan design, we compared DWMC with TimberWolf 1.3.3 [TW98] (which uses simulated annealing algorithm) and *relaxed simulated tempering* [CK99] (which is a variant of simulated tempering). For the comparison with TimberWolf, we did the experiments in two ways. First, we turn off TimberWolf's

routing space estimation ability to let TimberWolf generate solutions without routing space, as our floorplan tool currently does not reserve routing space.¹ Second, we allow TimberWolf to reserve routing space, but we then use a compaction tool to abut modules together in order to make a fair comparison. Table III shows the minimum area results and minimum wirelength results by running TimberWolf 25 times. In Table III, we also report the average runtime² of TimberWolf; for DWMC, two times are reported (in average for 5 runs): the time spent in DWMC (column *st*) and the overall runtime (column *overall*) (including DWMC time and the final simulated annealing time). Table III shows that DWMC results are better than those of TimberWolf's by up to 15.7% and 53.6% in area and wirelength.

The comparison with relaxed simulated tempering algorithm [CK99] is also favorable for DWMC: using comparable or less CPU time, DWMC achieves better results. For example, for the largest test case *playout*, DWMC used only about half of the CPU time needed by relaxed simulated tempering, and improved the wirelength result by 10%, while had a slightly worse (3%) area result.

To show the effectiveness of DWMC algorithm for constrained floorplan design, we compared the results with *simulated annealing*.³ In our test, we set k to 2, which means all the analog modules have to be placed into two analog regions. As shown in Table IV, DWMC algorithm reduces the wirelength by 19.5%, compared with simulated annealing algorithm.

Figure 2 shows the slicing floorplan solution obtained by using DWMC algorithm for circuit *playout*. We also show the floorplan result for the wireless modem in Figure 3.

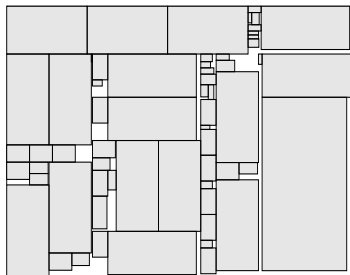


Fig. 2. A slicing floorplan obtained by using *dynamic weighting Monte Carlo* for circuit *playout*

¹TimberWolf allows the overlap of modules, if the compaction and global router are not activated in a placement refinement stage. In our test, we do compaction/de-compaction to remove any overlaps if they exist.

²CPU times are measured on a Sun ULTRA SPARC II (168MHz) workstation.

³TimberWolf cannot handle analog module constraints, so we implemented our own version of simulated annealing, which is shown to be efficient and effective [CK99].

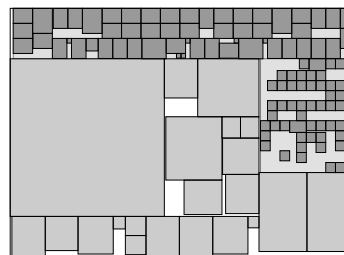


Fig. 3. A slicing floorplan obtained by using *dynamic weighting Monte Carlo* for a wireless modem

VI. CONCLUSION

This paper presents a new application of *dynamic weighting Monte Carlo* algorithm in VLSI floorplan designs. By introducing the importance weight as a dynamic variable, this approach allows the system to jump out of local minima more easily and thus sample better solutions. In this paper, we modified the Polish expression representation approach [WL89] to handle mixed signal MCM floorplan designs. Our experiments indicate that the dynamic weighting Monte Carlo algorithm is efficient and effective for both unconstrained and constrained floorplan designs. This is the first work adopting dynamic weighting Monte Carlo algorithm for solving optimization problems in the VLSI CAD field. We believe that this method has applications to many other VLSI CAD optimization problems.

REFERENCES

- [CK99] J. Cong, T. Kong, D. Xu, F. Liang, J. Liu, and W.-H. Wong, Simulated tempering for VLSI floorplan designs. In *Proc. Asia and South Pacific Design Automation Conf.*, pages 13-16, 1999.
- [GG84] S. Geman and D. Geman, Stochastic relaxation, Gibbs distributions and the Bayesian restoration of images. In *IEEE Transaction on Pattern Analysis and Machine Intelligence*, pages 721-741, 1984.
- [GT95] C.J. Geyer and E.A. Thompson, Annealing Markov chain Monte Carlo with applications to ancestral inference. In *Journal of the American Statistical Association*, pages 909-920, 1995.
- [KG83] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi Jr., Optimization by simulated annealing. In *Science*, pages 671-680, May 1983.
- [KR94] W. Kerler and P. Rehberg, Simulated tempering procedure for spin-glass simulations. In *Physical Review E*, pages 4220-4225, 1994.

TABLE III
COMPARISON OF DWMC ALGORITHM WITH TIMBERWOLF

circuit	TW with routing space & compaction			TW without routing space			DWMC			
	area(mm ²)	WL(μm)	time(sec)	area(mm ²)	WL(μm)	time(sec)	area(mm ²)	WL(μm)	st(sec)	overall(sec)
apte	49.74	525420	109.2	48.50	487710	66.0	48.50	226412	20.42	102.71
xerox	21.04	640660	148.6	22.64	526920	101.2	20.43	383909	20.05	152.62
hp	9.44	194840	133.0	9.58	186760	91.4	9.58	118472	22.90	128.45
ami33	1.33	62739	273.4	1.27	71800	221.0	1.29	45923	99.03	426.86
ami49	40.20	755960	544.8	40.81	814200	472.8	42.23	670845	225.90	922.94
playout	118.88	5286300	2968.3	115.52	6220900	1599.2	97.35	4575894	1128.80	3771.86

TABLE IV
COMPARISON OF SA AND DWMC FOR THE MIXED SIGNAL MCM DESIGN

circuit	Simulated Annealing			Dynamic Weighting Monte Carlo			Improvement(%)	
	area(mil ²)	wirelength(mil)	time(sec)	area(mil ²)	wirelength(mil)	overall(sec)	area	wirelength
modem	9.314 x 10 ⁵	6.481 x 10 ⁴	1388.12	9.312 x 10 ⁵	5.219 x 10 ⁴	8605.10	0.0	19.5

- [Li97] F. Liang, Weighted Markov chain Monte Carlo and optimization. *Ph.D. dissertation, the Chinese University of Hong Kong, 1997.*
- [LW99] F. Liang and W.H. Wong, Dynamic weighting in simulations of spin systems. In *Physics Letters A, 252*, pages 257-262, 1999.
- [MF95] H. Murata, K. Fujiyoshi, S. Nakatake, and Y. Kajitani, Rectangle-packing-based module placement. In *Proc. of Int. Conf. on Computer-Aided Design*, pages 472-479, 1995.
- [MP92] E. Marinari, and G. Parisi, Simulated tempering: a new Monte Carlo scheme. In *Europhysics Letters*, vol.19, (no.6), 15 July, 1992, pages 451-455.
- [NF96] S. Nakatake, K. Fujiyoshi, H. Murata, and Y. Kajitani, Module placement on BSG-structure and IC layout applications. In *Proc. of Int. Conf. on Computer-Aided Design*, pages 484-491, 1996.
- [Re95] G. Reinelt, TSPLIB 1995. In <http://ftp.iwr.uni-heidelberg.de/pub/tsplib>
- [SS94] J. Stander and B.W. Silverman, Temperature schedules for simulated annealing. In *Statistics and Computing*, pages 21-32, 1994.
- [PL93] P. Pan and C. L. Liu, Area minimization for general floorplans. In *Proc. of Int. Conf. on Computer-Aided Design*, pages 606-609, 1993.
- [TW98] TimberWolf Systems, Inc., TimberWolf Placement & Global routing software package. In <http://www2.twolf.com/benchmark.htm>
- [WL89] D.F. Wong and C. L. Liu, Floorplan design of VLSI circuits. In *Algorithmica*, pages 263-291, 1989.
- [WL97] W. H. Wong and F. Liang, Dynamic weighting in Monte Carlo and optimization. In *Proc. National Academic Science, USA*, pages 14220-14224, Dec 1997.
- [YT96] T. Yamanouchi, K. Tamakashi, and T. Kambe, Hybrid floorplanning based on partial clustering and module restructuring. In *Proc. of Int. Conf. on Computer-Aided Design*, pages 478-483, 1996.
- [YW97] F.Y. Young, and D.F. Wong, How good are slicing floorplans. In *Integration, the VLSI journal*, pages 61-73, 1997.