Statistical Power Profile Correlation for Realistic Thermal Estimation

Love Singhal* Sejong Oh± Eli Bozorgzadeh*

* Center for Embedded Computer Systems
University of California, Irvine, California 92697-3435
Email: {lsinghal,eli}@ics.uci.edu

± Korea Advanced Institute of Science and Technology
Daejeon 305-701, Republic of Korea
Email: sejong.oh.sayhi@gmail.com

***Abstract*— At system level, the on-chip temperature depends both on power density and the thermal coupling with the neighboring regions. The problem of finding the right set of input power profile(s) for accurate temperature estimation has not been studied. Considering only average or peak power density may lead either to underestimation or overestimation of the thermal crisis, respectively. To provide more realistic temperature estimation, we propose to incorporate multiple power profiles. Using the proposed statistical methods to determine the closeness between the power profiles, we apply a clustering algorithm to identify few input power profiles. We incorporate them in a thermal-aware floorplanner and empirical results show that using the single input power profile (average or peak) leads to 37% degradation in critical wire delay and 20% degradation in wire length, compared to using the multiple input power profiles.**

I. INTRODUCTION

Thermal-aware system level design flow requires modeling tools for estimation of temperature per unit area on a given power density profile [1]. Accurate thermal modeling techniques are computationally intensive. Hence, there is a large body of research work to provide efficient, yet accurate, thermal simulation in design flow. The temperature of each module is not only dependent on its own heat generation rate (i.e. its power density) but also dependent on heat coupling with neighboring modules. The physical layout of blocks (floorplan) plays a significant role on thermal behavior of the chip. Hence, many existing work on thermal estimation and analysis is on a given layout or toward generating a thermal-aware layout (*Thermal-aware Floorplanning*) [2]–[4]

However, all recently developed thermal-aware tools deploy temperature estimation techniques only on a single power profile representing power profiles of all inputs and all applications (e.g. using average or peak power profile). Different applications lead to different dynamic power profiles of the blocks. Most of the existing work use either average power or peak power per block of the applications for simulating temperature, without analyzing the impact of this assumption. Also, it is not practical to consider individual power profiles of each application in thermal estimation of a layout. The problem of finding the right set of input power profile(s) for accurate temperature estimation has not been studied. In this work, we present a study of why, how, and which power profiles should be used during the temperature estimation. Interestingly, only a recent work [5] shows that the power profile does not have major effect on the leakage power as long as the total power remains same. However, they do not consider the effect of power profile on temperature variation across different applications, especially the peak temperature of the blocks. Average power density [1] or peak power density cannot capture the relative power density variation among the blocks and hence, fails to provide realistic estimation of temperature.

This paper focuses on cluster analysis techniques to find a small set of power profiles for realistic thermal estimation of a given layout. We introduce a notion of *leader* power profile which represents the power profile of a subset of applications. Our contribution in this paper is as follows:

- We show that the single power profile like peak or average is not suitable for layout thermal estimation, and that few power leaders could adequately capture behavior of all applications
- We propose statistical distance metric to define closeness in power profiles for similar thermal behavior
- We develop a clustering algorithm to generate leader power profiles based on the statistical closeness metrics
- We show the integration of multiple power leaders in thermal and leakage aware floorplanner. By incorporating our proposed multiple leader profiles in our floorplanner, the floorplanner estimates the temperature of the blocks which are neither too pessimistic (unlike peak power profile), nor too optimistic (unlike average power profile). As a result, the floorplanner is leveraged for better optimization on wirelength and critical path delay.

Our work provides a new perspective to the temperature estimation approaches. We believe that this work will motivate the future design tools to consider multiple input power profiles for realistic temperature estimation.

II. EFFECT OF CHANGE IN POWER ON TEMPERATURE

A commonly used thermal model ($k\nabla^2 T + P = 0$) for steady state heat flow is given in [6]. In the equation, k is the thermal conductivity, T is the temperature, and P is the power density of heat sources. The thermal equation can be rewritten [4] in the matrix form as follows:

$$\mathbf{R}.\overrightarrow{P} = \overrightarrow{T} \tag{1}$$

where $\mathbf{R}$ is the thermal resistance matrix ($\mathbf{R_{ij}}$ is the thermal resistance between block i and block j), $\overrightarrow{P}$ is the power profile vector ($\overrightarrow{P}_i$ is the power dissipation of block i), and $\overrightarrow{T}$ is the temperature profile vector (T_i is the temperature of block i).

Using this model, we show that the error or change in the power of blocks affects the temperature of the blocks linearly. Consider a change in power of blocks $\Delta\overrightarrow{P}$, where ΔP_i represents change in power of block i. The change in the power of the blocks will result in the new temperature $\overrightarrow{T}'$, given by $\overrightarrow{T}' = \mathbf{R}.(\overrightarrow{P} + \mathbf{\Delta}\overrightarrow{P})$ which is equal to $\mathbf{R}.\overrightarrow{P} + \mathbf{R}.\mathbf{\Delta}\overrightarrow{P}$. Hence, the change in the temperature of the blocks $\Delta\overrightarrow{T}$ is given by $\Delta\overrightarrow{T} = \mathbf{R}.\mathbf{\Delta}\overrightarrow{P}$. Thus the change in the temperature is linearly dependent on the amount of change in the power.

Lemma 1: The change in temperature due to the change in power ($\Delta\overrightarrow{P}$) that is not dependent on temperature, is a linear function of $\Delta\overrightarrow{P}$.

Dynamic power consumption of the microarchitectural blocks is not dependent on the temperature. Hence, any increase/decrease in dynamic power leads to linear increase/decrease in the temperature of the blocks. However, in nanoscale technology, static power, which is temperature-dependent, cannot be neglected and can have a significant impact on temperature of the chip especially at higher temperature.

[1] In this paper, we use power profile and power density interchangeably.

978-1-4244-1922-7/08/$25.00 ©2008 IEEE

Leakage power is an exponential function of temperature ($P_L(\overrightarrow{T})$). The initial temperature is computed as: $R.(\overrightarrow{P} + \overrightarrow{P_L}(\overrightarrow{T_0}))$. $\overrightarrow{T_0}$ is the steady state temperature before $\Delta\overrightarrow{P}$ occurs. Since, leakage power depends on the temperature, the value of leakage power changes at the new temperature. This positive loop of temperature computation and leakage power update continues until the design reaches steady state temperature (or otherwise it goes to *thermal runaway*). Assume after k iterations, the steady state temperature, $\overrightarrow{T_k}$ is reached, which can be formulated as:

$$\overrightarrow{T_k} = R.(\overrightarrow{P} + \Delta\overrightarrow{P} + \overrightarrow{P_L}(T_k)) \quad (2)$$

Theorem 1: The $\Delta\overrightarrow{P}$ change in the dynamic power leads to linear increase in the steady state temperature as well as additional non-linear increase in temperature due to the presence of temperature-variant leakage power.

III. Power Profiles in Temperature Estimation

In order to fairly estimate the temperature of functional blocks and perform a complete thermal simulation of a general purpose or an application specific processor, designers use a set of benchmark applications to run on the processor. These benchmark applications usually cover all the possible scenarios of the usage of each functional blocks - from heavy usage of certain functional blocks to the medium usage of all the functional blocks. Designers provide either single application, or or average/peak power numbers of all the benchmark applications to the temperature estimators in design tools. However, actual power numbers of some applications could be very different from the average or peak power numbers.

A leader power profile that represents power values of multiple applications running on a chip will have some errors in the power values compared to the actual values of the applications. For an application m, the leader profile will have $\Delta\overrightarrow{P}_m$ error, such that $\overrightarrow{P}_l = \overrightarrow{P}_m + \Delta\overrightarrow{P}_m$, where $\overrightarrow{P}_l$ is the leader power profile. This will lead to an error in the temperature of the leader profile which is given by $\Delta\overrightarrow{T}_m = \mathbf{R}.\mathbf{\Delta}\overrightarrow{P}_\mathbf{m}$. Hence, representing all power profiles with single leader power value per module leads to inaccuracy in temperature estimation. In the next subsections, we discuss the impact of $\Delta\overrightarrow{P}$ imposed by widely used *average* power and *peak* power profiles.

A. Average Power Leader

In average power, the power values of each block are averaged over all the benchmark applications. The average, however, levels any extreme variation in the power of blocks that may occur in some applications. Since those applications *could* run on the processor, they will have higher temperature than estimated. In the experimental section, we show the errors generated by average power profile.

B. Peak Power Leader

In the *peak* power profile, the maximum power density of each block over all the benchmark applications is assigned as the power density of the block. The peak power considers the *worst-case* scenario of the power dissipation of each block. Since it considers the highest power dissipation of all the blocks, it finds the highest temperature of each block over all the applications. Normally, this is useful as most of the design tools are interested in knowing the worst case temperature. However, the estimated temperature of the whole chip could be significantly higher than achieved in any single application.

Since, neither of the two single power leaders - *average* and *peak* - can accurately predict temperatures, we need a new metric to define how close leader power profiles are to the actual power values of the applications.

IV. Power-Profile Correlation

Before we find a leader of two or more applications, we must first identify subset of the applications that can be represented by a single leader. Each application has a different power profile, which means that the power consumed by each block on the chip is different for different applications. If the power vectors of two or more applications show similar behavior (same highs and same lows), they will have similar thermal behavior with any given layout. Two applications are said to be **close** to each other if they have similar thermal behavior on any given floorplan.

The notation of the power numbers of two applications in this paper is $X =< X_1, X_2, \cdots, X_k >$ and $Y =< Y_1, Y_2, \cdots, Y_k >$ for the k blocks on the chip. In cluster analysis techniques [7], several distance metrics are provided to measure distance/similarity among a set of data. In this work, we give a closeness metric to mathematically quantify similarity between the power profiles of any two applications. Our closeness metric is intuitively suited to our case keeping in mind the relationship between power and temperature.

A. Correlation Metric

The correlation metric calculates the statistical correlation of two applications over different blocks. The random variables are the power numbers of the blocks in the two applications, X and Y. The power of block i in first application is X_i. Let the first moment (average) of the random variable X be $\overline{X}$ and Y be $\overline{Y}$. First, we evaluate covariance of the two applications which is defined as $cov(X,Y) = E[(X - \overline{X})(Y - \overline{Y})]$ where E is the expected value of the product.

$$cov(X,Y) = \frac{1}{k} \times \sum_{i=0}^{k}[(X_i - \overline{X}) * (Y_i - \overline{Y})] \quad (3)$$

The covariance of the two random variables will calculate how similarly the power of two applications vary with each other from their respective means. Covariance will be high if the two appplications have similar highs and similar lows in their power profile, and covariance will be low if the two applications have opposite highs and lows. The covariance of any two applications could be any real number. We use normalized form of the covariance, which is called as *correlation*, defined as:

$$\rho_{XY} = \frac{cov(X,Y)}{\sigma_X * \sigma_Y} \quad (4)$$

The correlation metric has some important properties. The correlation of any two applications is bounded, and $-1 \leq \rho \leq 1$. Further, when $Y = \alpha X + c$, then ρ_{XY} is equal to 1, where α is any positive real number and c is any real constant. That is, when Y is a linear function of X, then the two variables have the highest correlation. Hence, correlation is a strong metric to define closeness of two applications. We now define a distance function d_ρ between any two applications X and Y using equation 5.

$$d_\rho(X,Y) = 1 - \rho_{XY} \quad (5)$$

The distance d_ρ is a positive real number where $0 \leq d_\rho \leq 2$. Also, $d_\rho(X,Y) = d_\rho(Y,X)$ for all X and Y.

V. Power Profile Clustering

In order to find the leaders of a set of benchmark applications, we should cluster applications that are close to each other in a few sets. We propose the following objective for our problem.

Problem Statement: Given a positive integer k, find a clustering of n applications such that there are k clusters and the maximum distance, d_{max}, between any two nodes of a cluster is minimized.

Each such cluster can then be represented using a leader during floorplanning. The high value of k will lead to low value of the maximum distance d_{max}, and hence, lead to clusters with highly correlated applications. If $k = n$, then all nodes are in separate clusters, representing a solution with the most accurate representation

($d_{max} = 0$). This clustering problem is an NP Hard problem. [7] provides a good discussion of various clustering techniques. In this work, we propose a hierarchical bottom-up clustering algorithm. The pseudo code of the clustering algorithm is shown in Algorithm 1. Before clustering, we remove redundant applications from the given set of applications. The next subsection discusses this important aspect of the clustering of power profiles.

Algorithm 1 K-Clustering Algorithm for minimizing d_{max}

```
1: function K-CLUSTERING (k)
2:     Remove all redundant applications.
3:     Sort all the edges in the increasing order.
4:     Start from the least weighted edge.
5:     repeat
6:         Pick the next minimum weight edge, e
7:         c1 = cluster containing one node of e
8:         c2 = cluster containing the other node of e
9:         /* Let wt: E → ℜ, the weight of edge, e */
10:        /* Let interedge: C × C → E*,
11:          set of edges between two given clusters */
12:        if wt(e') ≤ wt(e), ∀ e' ∈ interedge(c1, c2) then
13:            merge c1 and c2
14:        end if
15:    until the number of clusters > k
16: return wt(e)
17: end function
```

A. K-Clustering Algorithm

After removal of redundant applications, we form a complete graph of all the remaining applications. Each benchmark application is created as a node and put inside a cluster. The distance between any pair of nodes is precomputed and is taken as the weight of an edge between the two nodes.

Algorithm 1 creates clusters in a bottom-up manner by starting from very small size clusters (one node each) and merging the clusters until the desired number of clusters is reached. The algorithm first sorts each edge by distance. Since we want to minimize the maximum distance between the nodes of a cluster, we start by merging the clusters with very low weight edges. Therefore, we first pick the lowest weight edge. While merging any two clusters, we ensure that no edge inside the new cluster has weight higher than the current weight. This guarantees that the last edge picked by the algorithm is the highest weight edge inside any cluster. If we cannot merge the clusters with the current edge, we move to the next edge in the sorted list. The next edge will have more chance to merge two clusters than the previous edge because its weight is higher than the previous edge. Since the graph is a complete graph, the algorithm will eventually terminate before all the edges are visited. Only in the case of $k = 1$, the algorithm terminates at the last edge.

Theorem 2: The time complexity of the K-clustering algorithm is $O(nlogn + m^2)$ where m is the number of edges.

We choose the distance function defined in Section IV to compute the various clusters. After clustering, we create a leader for each cluster. The leader of a cluster is calculated as the peak (maximum) power values of all the applications inside the cluster.

VI. Thermal Aware Floorplanning for Multiple Applications

We develop a thermal aware floorplanner that uses multiple power profiles for accurate temperature estimation. Our thermal aware floorplanner is developed from HotFloorplan [3] tool which is an open source thermal aware floorplanner widely used in the research for microarchitectural design. It is a simulated annealing based floorplanner which reduces the linear combination of area, wirelength and peak temperature of the device. It takes a single power profile for temperature estimation. The moves of the simulated annealing engine are the soft blocks moves (changing the shape of blocks) and the swapping moves (swapping positions of blocks). Our floorplanner has the following additions to HotFloorplan:

1) It takes multiple leader power profiles as input, calculates temperature of each profile, and finds the peak temperature over all profiles, during simulated annealing iterations. Thus, our floorplanner can find more accurate temperatures during simulated annealing. The clustering algorithm described in Section V is used to reduce the number of leaders. After the applications are clustered, a representative power profile from each cluster can be used by the floorplanner to simulate the steady state temperatures.
2) It handles leakage power and the positive thermal feedback loop of the leakage power to compute steady state temperatures during simulated annealing iterations. The thermal feedback loop is stopped when the change in temperature becomes very small. In order to avoid complex floating point leakage power calculations and reduce runtime of the floorplannner, we store the leakage power values of each block at each temperature in a look-up table. The table is computed offline.
3) It considers thermal runaway temperature of the chip. In [8], a thermal runaway condition is described based on the positive feedback loop of the leakage power and temperature. If the package's heat removal ability is not adequate, it can lead to thermal runaway and catastrophic heat failures. The lowest temperature that meets criteria of infinite feedback loop is called *runaway temperature*. Our floorplanner takes thermal runaway or thermal threshold temperature as an input and rejects any floorplan in which this temperature is reached during simulated annealing iterations.
4) It includes wire delay in the cost function.

VII. Experiments

A. Experimental Setup

Figure 1 illustrates an overview of the proposed design flow. The process of temperature estimation for each leader profile is shown in the right side of Figure 1. We evaluated our floorplanner on Alpha 21264 processor. To target the processor, the SimpleScalar's out of order instruction simulator and Wattch are extended, breaking down the RUU and functional units of the sim-outorder into functional blocks on the Alpha processor. We also augmented the sim-outorder to obtain the power profiles of the blocks with an interval of 10k instructions. The sim-outorder and the power model are configured similar to Hotspot in [9]. The configurations of power model are 0.70nm, V_{dd} = 1.0V and a clock speed of 3 GHz. To obtain the dynamic power profiles of the blocks, we ran 17 integer and 19 floating-point applications from the SPEC CPU2000 benchmark suit. Each benchmark is run with one reference input or one test input for 500 Million instructions which follow architecture and thermal warmup of 300 Million instructions like [3]. We find that 21 benchmarks from the reference set and 16 benchmarks from the test set generate the power trace in this interval.

We use HotLeakage [10] to compute the leakage power of blocks. HotLeakage is configured with the same configuration file as the SimpleScalar. The ambient temperature is 333.15K and the thermal threshold temperature is chosen to be 366K. Though the current thermal runaway temperature at 70nm is higher than 366°K, we choose this temperature as a threshold in order to simulate thermal runaway and extreme thermal bottlenecks within the range of the power values in the generated power profiles in this setup. In general, the thermal runaway temperature is expected to decrease with increasing clock frequency to lower levels. All the experiments are performed on Linux machines with Dual Intel Xeon 2.0 GHz processors and 1GB RAM each.

B. Comparison with Peak and Average Power Leaders

All the current thermal-aware floorplanners consider only a single leader power profile found by computing the average power values [2] [4] (referred to as *average-leader* floorplanner) or peak power values

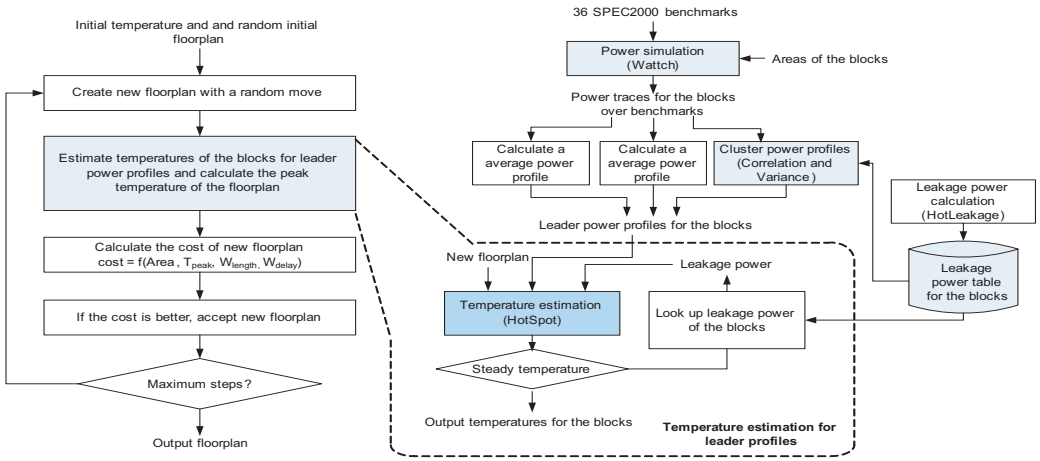

Fig. 1. Thermal/Leakage/Power-Profile Aware Floorplanning Design flow

of all benchmark applications (referred to as *peak-leader* flooplanner). We compare these with n-leader floorplan in which all applications represent themselves as the leaders (n-*leader* floorplanner). Table I shows the results of the three floorplans generated using the peak power, average power and n-leader power profiles. It shows the critical path wire delay, wire length, and the peak temperature of the floorplan. The peak temperature of the floorplan is computed by finding the steady state temperatures of all the applications on the floorplan using HotSpot.

Floorplan Type	Maximum Wire Delay	Wire length	Peak temperature (in Kelvin)
$k = n$	**1.658**	**0.040**	363.5
Peak	2.276	0.048	363.9
Average	2.130	0.045	**366**

TABLE I

COMPARISON BETWEEN THE THREE FLOORPLANS GENERATED BY THREE METHODS OF GENERATING LEADER POWER PROFILES.

In Table I, the n-leader floorplan is significantly better both in wirelength and in wire delay. It has 37% better wire delay and 20% better wirelength compared to the peak-leader floorplan. The area and maximum temperature of the two floorplans are the same. Since the peak-leader floorplan overestimates the temperature significantly, it reaches the thermal threshold in most of the simulated annealing iterations even though none of the applications may actually reach this threshold. Thus, many floorplans are rejected by the floorplanner. Since the n-leader floorplanner has the actual temperatures of the blocks in all applications, it does not reject those floorplans. Thus using n leaders significantly improve the quality of the floorplan. This is, however, an exhaustive solution and leads to high runtime. The runtime of n-leader floorplanner is 2.2 times the runtime of average-leader or peak-leader floorplanner. The results in Table I show that the average-leader floorplan crosses the thermal threshold. The floorplanner misses the hotspots occurring on the floating point units in most of the cases during simulated annealing. Hence, average-leader power profile as mostly used in thermal-aware floorplanning does not consider the worst possible cases of the temperatures.

C. Results on Distance Metric Based Clustering

In k-leader floorplanning refers to our proposed thermal-aware floorplanner with k leader prower profiles (Section VI). Table II shows the result of k-leader floorplanning using correlation metric for clustering. Speedup refers to the improvement in runtime over n-leader floorplanning. Number of clusters in the tables refers to the number of leader power profiles. In general, if the number of clusters (k) decreases, the quality of the floorplan (wirelength and wire delay) decreases. For example, in Table II, the wire delay of the floorplan when $k = 15$, is 1.656, whereas it is 1.804 for the floorplan when $k = 2$. As the number of clusters decreases, error inside each cluster also increases and causes higher temperature error. Also, the results from clustering methods are in general better than the average-leader and peak-leader floorplanners as shown in Table I. We can find the overall good solutions at the low cluster counts of 2 and 3 with a speedup of around 3 times compared to the runtime of the exhaustive search ($k = n$ leaders).

No. of Clusters (k)	Maximum Wire Delay	Wire Length	Speedup	Peak temperature (in Kelvin)
15	1.656	0.038	1.61	361.7
6	1.761	0.046	2.28	363.9
3	1.83	0.038	2.9	363.7
2	1.804	0.043	2.9	362.8

TABLE II

k-LEADER FLOORPLANNER USING CORRELATION DISTANCE METRIC d_ρ.

VIII. CONCLUSION

Thermal-aware system design tools estimate the temperature of the chip using just a single power profile (average or peak). However, these single power profiles fail to identify or over-estimate the hotspots of the chip. We therefore need suitable leader power profiles which can be used to predict the steady state temperature of all the applications. In this paper, we propose a cluster analysis technique to group the correlated power profiles in order to provide the tools with more realistic temperature values in an efficient manner. The experimental results show that using only a single power profile could increase wire delays of floorplan by 37%.

REFERENCES

[1] K. Skadron *et al.*, "Temperature-aware microarchitecture," in *Proceedings of ISCA*, June 2003, pp. 2–13.
[2] A. Gupta *et al.*, "Leaf: A system level leakage-aware floorplanner for socs," in *Proc. of ASPDAC*, 2007, pp. 274–279.
[3] K. Sankaranarayanan *et al.*, "A case for thermal-aware floorplanning at the microarchitectural level," *Journal of Instruction-Level Parallelism*, vol. 7, no. 10, Oct. 2006.
[4] M. Healy *et al.*, "Microarchitectural floorplanning under performance and thermal tradeoff," in *Proc. of DATE*, 2006, pp. 1288–1293.
[5] Y. Liu *et al.*, "Accurate temperature-dependent integrated circuit leakage power estimation is easy," in *Proc. of DATE*, 2007, pp. 1526–1531.
[6] C. Tsai and S. Kang, "Cell-level placement for improving substrate thermal distribution," *IEEE TCAD*, vol. 19, no. 2, pp. 253–266, 2000.
[7] H. C. Romesburg, *Cluster Analysis for Researchers*. Lifetime Learning Publications, 1984.
[8] L. He *et al.*, "System level leakage reduction considering the interdependence of temperature and leakage," in *Proc. of DAC*, 2004, pp. 12–17.
[9] K. Skadron *et al.*, "Temperature-aware microarchitecture: Modeling and implementation," *ACM TACO*, vol. 1, no. 1, pp. 94–125, 2004.
[10] Y. Zhang *et al.*, "Hotleakage: A temperature-aware model of subthreshold and gate leakage for architects," Univ. of Virginia Dept. of Computer Science Technical Report CS-2003-05, Tech. Rep., Mar 2003.