University of Minho
Department of Informatics

Master's Degree in Informatics Engineering
Parallel and Distributed Computing

**Numerical Methods and Algorithms**

# Simulated Annealing

## The Room Assignment Problem

Group 1
Miguel Palhas — pg19808@alunos.uminho.pt
Pedro Costa — pg19830@alunos.uminho.pt

Braga, May 2012

## 1 Introduction

Physical annealing is a process through which a melted metal is allowed to cool slowly, thus forming a defect-free crystal with a regular structure. At higher temperatures the atoms move more freely and rearrange their structure easily. As the temperature decreases, the ability for the atoms to move is diminished since they possess less energy. This allows the material to reach a state of minimum energy – the crystalline form.

Simulated annealing is an iterative method based on the process of physical annealing for solving combinatorial optimization problems. The system starts at a given (high) "temperature", which allows the algorithm to choose and follow a worse solution according to a probability function based on the "temperature". In each iteration, a cooling function decreases the system "temperature", also decreasing the probability for a worse solution to be followed. Without simulated annealing, the algorithm never accepts a worse solution, which causes it to be trapped in local solutions. With simulated annealing, the algorithm is allowed to change the system state from a local solution to a worse state, which eventually leads to a state closer to the global solution.

The room assignment problem aims to assign $n$ students to $\frac{n}{2}$ rooms, so that the probability of interpersonal conflicts is minimized. The probability of a conflict is given by a matrix $D$ of $n^2$ integer values where each element $d_{ij}$ represents the probability for a conflict between the students $i$ and $j$ (higher values mean higher chances of a conflict occurring).

In this project two versions are implemented to solve this problem: the first being the sequential implementation and the second being a parallel implementation using MPI. Both versions implement an approach using a strictly minimizing approach and another using the simulated annealing method. For each version, both approaches are tested against the same matrix of incompatibility values for a range of different numbers of students to be assigned. The versions are also tested using different initial temperature values in the approach using simulated annealing.

## 2 The Room Assignment Problem

The room assignment problem is a combinatorial optimization problem where the simulated annealing method allows better solutions to be achieved.

The problem consists in assigning $n$ students to $\frac{n}{2}$ rooms, but in a way that minimizes the probability of a conflict occurring. The chances for such conflicts to occur are measured in a matrix $D$ of $n^2$ integer values, where each element $d_{i,j} = d_{j,i}$ represents how much students $i$ and $j$ dislike each other. In any iteration, the current system state is represented by a vector $S$ of $\frac{n}{2}$ pairs of elements, where each pair $(i, j)$ represents the two students assigned to a room[1]. The total cost of a state $S$ is given by the sum of all the $d_{i,j}$, where $(i, j) \in S$. A formal definition of this problem as an optimization problem is presented in figure 1.

**Decision Variables**

- $d_{i,j}$ — level of incompatibility between the students $i$ and $j$;

**Decision Restrictions**

- $0 \leq i \leq n$
- $0 \leq j \leq n$
- $\forall i, j : 0 \leq d_{i,j} \leq 10$

**Objective Function**

$$\min\ :\ \mathbf{Cost}(S) = \sum \{d_{i,j} : (i,j) \in S\} \tag{1}$$

Figure 1: Formal definition of the Room Assingment Problem as an optimization problem.

The goal in this problem is to find a state for which the cost is minimized. However, for $n$ students there are $(n-1) \times (n-3) \times \ldots \times 3$ possible distributions – this has the order of magnitude of $\sqrt{n!}$, which makes it unfeasible to analyze every single possible state. Alternatively, with an heuristic method, while it does not guarantee the optimal solution, it allows to calculate an approximation in significantly less time.

The heuristic approach to this problem starts by generating a random room distribution. In each iteration, two students in distinct rooms are randomly selected and swapped. If the new state has a lower cost, the change is accepted, otherwise it is reversed.

The simulated annealing method changes how the decision about accepting or refusing the swap is taken. With no simulated annealing, only a state with lower cost is accepted. This causes the system to be trapped in local minima. With simulated annealing, the system is initialized at a given (high)

[1] Since $d_{i,j} = d_{j,i}$ then $(i,j) \in S \Leftrightarrow (j,i) \in S$. In other words, the order in the pair is neglected.

temperature and a higher cost state is accepted with probability $e^{-\Delta/T}$, where $\Delta$ is the difference between the cost of the two states (before and after the swap). A cooling function is used in each iteration to (slowly) decrease the temperature.

## 3 Implementation

### 3.1 Parallelism

## 4 Experimental Comparison

### 4.1 Environmental Setup

All the tests were performed in the SeARCH cluster[2] using a specific group of computational nodes, referred in this document as SeARCH Group Hex. This group contains six nodes, each with two hex-core processors and up to 48 GB of RAM. The detailed hardware description for the nodes in this group is presented in table 1.

| | |
|---|---|
| Processors per node: | 2 |
| Processor model: | Intel®Xeon®X5650 |
| Cores per processor: | 6 |
| Threads per core: | 2 |
| Clock frequency: | 2.66 GHz |
| L1 cache: | 32 KB + 32 KB per core |
| L2 cache: | 256 KB per core |
| L3 cache: | 12 MB shared |
| RAM: | 12 to 48 GB |

Table 1: SeARCH Group Hex hardware description. See [xeo, 2011] for further detail about this processor.

### 4.2 Methodology

Each implementation described in this document was tested for different values of $n$ from 40 to 200 with a step of 4. The parallel implementation with MPI was also tested for the initial temperature of 1.0 and 10.0.

For each set of parameters, only one execution was performed.

### 4.3 Results

## 5 Conclusion

## References

[xeo, 2011] (2011). Intel® Xeon® Processor 5600 Series Datasheet Volume 1. Intel Corporation. Revision 002.

[2] http://search.di.uminho.pt