A COMPARATIVE STUDY OF ANNEALING TECHNIQUES FOR ACADEMIC COURSE SCHEDULING

Saleh Elmohamed
Geoffrey C. Fox
Paul Coddington

Northeast Parallel Architectures Center
Syracuse University
Syracuse, NY 13244

August 20, 1997

A Comparative Study of Annealing Techniques for Academic Course Scheduling

Saleh Elmohamed, Geoffrey Fox, Paul Coddington

Northeast Parallel Architectures Center
Syracuse, NY 13244
USA

{saleh, gcf, paulc}@npac.syr.edu

Academic Scheduling at the University Level

About the Problem:

In general, the problem of timetabling is the assignment of timeslots to a set of events *subject to* a set of constraints (hard, medium, soft).

Main Components:

courses N_q, rooms N_x, buildings N_b, departments N_d, students N_s, professors N_p, pairs of professor-classes N_c, and timeslots N_t.

The Objective:

Given these components and a set of constraints, the objective is to construct a legal schedule.

Methods Used

1. Mean-field annealing (MFA).
2. Simulated Annealing (SA) using geometric cooling.
3. SA using adaptive cooling.
4. SA using adaptive cooling and reheating as a function of cost.
5. SA using cooling schedules (2), (3), and (4); and a rule-based expert system as a preprocessor.

Mean Field Annealing

Summary:

It is a deterministic approximation method to SA, by attempting to average over the statistics of the annealing process.

Pluses:

Execution/simulation time is much faster than SA (using any cooling schedule).

Minues:

For timetabling, solution quality is no better than solutions obtained by SA. Depending on the size and constraints dealt with, quite often it is poorer ...

Problem Mapping in Mean Field Annealing

- Assume we have a Feedback Neural Net.
- An event-based mapping is carried out as follows (a sample):
  - event(i): prof-class(p,q) $\longrightarrow$ state(a): room-timeslot(x,t) represented by the following Potts neuron:

$$S_{pq;xt} = \begin{cases} 1 & \text{if event } (p, q) \text{ occur in space-time } (x, t) \\ 0 & \text{otherwise} \end{cases}$$

    provided

$$\sum_a S_{ia} = 1$$

  - The constraints involved are embedded in the network in term of weights W_{lk} between the neurons.

The main concept of MFA

- Approximate the energy function E by $E^{'}$. E is a function of discrete neural variables, while $E^{'}$ is represented by continuous MFA variables U_{ia} and V_{ia}.
- V_{ia} is an approximation to the average value of S_{ia} at temperature T.
- The heart of the method is a couple of MFT equation:
  1. $U_{ia} = -\frac{1}{T}\frac{\partial E}{\partial V_{ia}}$
  2. $V_{ia} = \frac{e^{U_{ia}}}{\sum_b e^{U_{ib}}}$

Solution of the MFA equations

- Equations (1) and (2) above can be solved iteratively using either synchronous or serial updating.
- T_c , the critical temperature, is used to set the scale of T. T_c is computed by expanding equation (2) around some fixed-point such as $V_{ia}^{(0)} = \frac{1}{N_a}$, where N_a is the number of possible states of each neuron.
- Any solution need to satisfy $\sum_a V_{ia} = 1 \quad \forall\ i$

Simulated Annealing

Let $\mathcal{N}_s$ be the neighborhood of configuration s. k is the Boltzmann constant = 1.

1. Generate an initial solution (configuration) s .
2. Compute cost of s : $C(s)$.
3. Compute initial temperature T_0.
4. Set the temperature $T = T_0$.
5. While stop criterion is not satisfied do:
   5.1. Repeat Markov chain length (M) times:
      5.1.1. Select a configuration $s' \subset \mathcal{N}_s$.
      5.1.2. Let $\Delta = C(s') - C(s)$.
      5.1.3. If ($\Delta \leq 0$ {downhill move}), set $s = s'$.
      5.1.4. If ($\Delta > 0$ {uphill move}), set $s = s'$ with a probability $= e^{-\Delta/(kT)}$.
   5.2. Reduce (or update) temperature T.
6. Return the solution s.

Mapping of Timetabling Components into SA

The three major components of relevance: **State**, **Cost**, and **Swap**.

State: is a timetable containing N_q, N_x, N_p, N_t, and N_s.

Cost: or energy is $E(N_p) + E(N_x) + E(N_q) + E(N_s) + E(N_t) + \ldots$

Swap: is the exchange of one or more classes, i.e. class c_i with class c_j w.r.t time/space, etc.

Annealing Schedule: Geometric Cooling

The new temperature in the above schedule is usually calculated according to:

$$T' = \alpha T \quad ,$$

where α is typically in the range $0.95 \ldots 0.99$. This schedule is used as a base line for comparison of other scheduling schemes.

Advantages

- Have a good theoretical foundation.
- Proven history of being good in practice.
- Well understood by all researchers working on annealing.

Disadvantages

Does not take into account the state of the system in any way, and thus can not adapt the intensity of the search depending on the difficulty scale or complexity of the problem.

Annealing Schedule: Reheating As a Function of Cost (RFC)

Some Background

- In combinatorial optimization problems, **phase transitions** can be observed as sub-parts of the problem are resolved. One observes that:
  1. At high temperatures, the gross structure of the solution is resolved.
  2. At low temperatures, the minute details of the solution are resolved.
- Applying a reheating type procedure, depending on the phase, would allow the algorithm to spend more time in the low temperature phases, thus reducing the total amount of time required to solve a given problem.
- To calculate the temperature at which a **phase transition** occurs, it is necessary to compute the **specific heat** of the substance. A **phase transition** occurs at a temperature T_{msp}, when the specific heat is maximal, and this triggers the change in the state ordering.

RFC - The Schedule

Due to Abramson at al., Kirkpatrick et al., and White and consists of:

1. Let C_i to denote the cost of configuration i, and $k = 1$ (the Boltzmann factor).
2. At T, generate configurations with a Boltzmann probability distribution: $P_i(T) = \frac{e^{\frac{-C_i}{kT}}}{\sum_j e^{\frac{-C_j}{kT}}}$
3. For a set $\mathcal{C}$ of $\mathcal{N}_T$ configurations at T, from which an estimate $< C(T) >$ of the average cost can be computed using: $< C(T) >= \sum_{i \in \mathcal{C}} C_i / \mathcal{N}_T$
4. Similarly, the average square cost at T: $< C^2(T) >= \sum_{i \in \mathcal{C}} C_i^2 / \mathcal{N}_T$
5. The variance of the cost at T: $\sigma^2(T) =< C^2(T) > - < C(T) >^2$
6. Now, the specific heat is: $C_H(T) = \frac{\sigma^2(T)}{T^2}$
7. Compute $T(C_H^{max})$ at which the max specific heat occurs, or at which the system undergoes a phase transition.
8. Compute the new temperature: $T_{new} = K \cdot C_b + T(C_H^{max})$ where K is a tunable parameter and C_b is the current best cost.

More Observations on RFC Schedule

- Reheating is done when T drops below the **phase transition** and no decrease in cost for a specified number of iterations (i.e. stuck in local minimum).
- Reheating increases the temperature above the phase transition, in order to produce enough of a change in the configuration to allow it to explore other minima when T is reduced again.
- If the best solution found so far is poor (i.e. has a high cost) then the super-structure may require re-arrangement. This can be done by raising the temperature to a level which is higher than the **phase transition** temperature (T_{msp}). Generally, the higher the best cost so far, the higher the temperature which is required to escape the local minimum.

{saleh, gcf, paulc}@npac.syr.edu

Annealing Schedule: Adaptive Cooling

- New T is computed based on the standard deviation of all costs obtained at the current T, i.e. the specific heat.
- **The idea**: keep the system close to equilibrium, by cooling slower close to the phase transition, where the specific heat is large.

Adaptive Cooling: The Schedule

1. Let T_j denote the current temperature, compute the standard deviation $\sigma(T_j)$ as outlined in RFC.

2. The new temperature T_{j+1}: $T_{j+1} = T_j \cdot e^{-\frac{aT_j}{\bar{\sigma}(T_j)}}$ where a is a tunable parameter.

3. Following suggestions by Otten et al., Diekmann et al., $\sigma(T_j)$ is smoothed out in order to avoid any dependencies of the temperature decrement on statistical fluctuations in the measurement of the standard deviation σ. $\bar{\sigma}(T_{j+1}) = (1-\omega)\sigma(T_{j+1}) + \omega\sigma(T_j)\,, \frac{T_{j+1}}{T_j}$ where we have set ω to 0.95.

4. The smoothing process preserves the key relationship: $\frac{d}{dT}C(T) = \frac{\bar{\sigma}^2(T)}{T^2} = C_H$

Few Key Strategies on Moves

- Limit swapping to categories having objects in common.
- Prioritize scheduling of classes based on, for example, size, enrollment, preference, etc. So swapping is done based only on those categories.
- Always, if possible, use key hints or relationships to make the swaps.
- In any multiphase approach to the problem like ours, it is quite essential to preserve as much as possible of the results obtained from the initial stages (or preprocessing).

Some Experimental Results

Sparseness of the Problem: The ratio is computed by $(N_x N_t - N_c)/(N_x N_t)$. Dense problems with relatively few spare space-time slots have a low **sparseness** ratio.

Academic Time Period	Sparseness Ratio
First Semester	0.50
Second Semester	0.53
Third Semester	0.62

{saleh, gcf, paulc}@npac.syr.edu

Academic Term	Algorithm	Sched. (avg) %	highest Sched. %	lowest Sched. %
1st Term	SA (geometric)	65.00	67.50	56.80
	SA (adaptive)	67.80	70.15	61.20
	SA (cost-based)	70.20	72.28	68.80
	ES	76.65	76.65	76.65
	MFA	65.60	71.00	61.00
2nd Term	SA (geometric)	65.65	68.00	57.10
	SA (adaptive)	68.50	70.10	60.77
	SA (cost-based)	75.14	77.68	70.82
	ES	79.00	79.00	79.00
	MFA	67.20	75.00	65.00
3rd Term	SA (geometric)	83.10	86.44	68.50
	SA (adaptive)	85.80	89.00	70.75
	SA (cost-based)	91.20	95.18	85.00
	ES	96.80	96.80	96.80
	MFA	88.00	95.00	82.00

Table 1: No preprocessor was used with the three methods.

Academic Time Period	Algorithm	Scheduled (average) %	highest Scheduled %	lowest Scheduled %
First Semester	SA (geometric)	93.90	95.12	85.20
	SA (adaptive)	98.80	99.20	95.00
	SA (cost-based)	100.0	100.0	100.0
Second Semester	SA (geometric)	95.00	98.95	89.40
	SA (adaptive)	99.00	99.50	98.50
	SA (cost-based)	100.0	100.0	100.0
Third Semester	SA (geometric)	97.60	98.88	90.90
	SA (adaptive)	100.0	100.0	100.0
	SA (cost-based)	100.0	100.0	100.0

Table 2: A rule-based system was used as preprocessor.