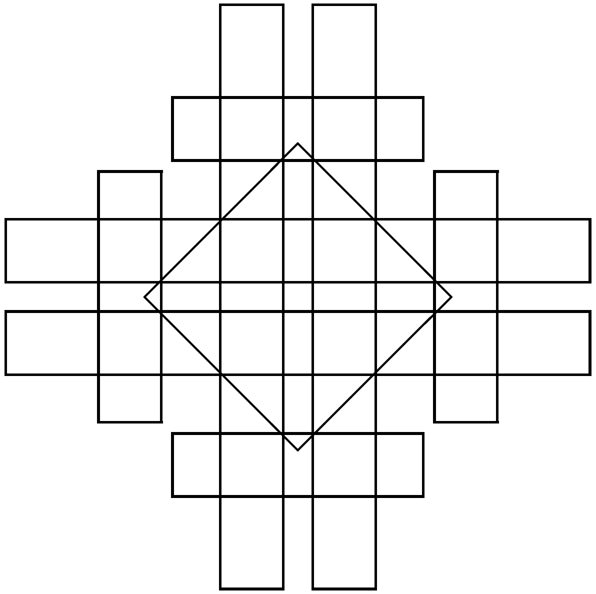
OPTIMIZATION TECHNIQUES FOR ACADEMIC SCHEDULING AND TIMETABLING

Saleh Elmohamed
Geoffrey C. Fox
Paul Coddington

Northeast Parallel Architectures Center
Syracuse University
Syracuse, NY 13244

September 21, 2000



Saleh Elmohamed NPAC, saleh@npac.syr.edu

Academic Scheduling at the University Level

Definition

For a certain school, given N_p teachers, N_q classes, N_x classrooms or lecture halls, and N_s students; it is required to schedule N_l lessons (defined as teacher-class pairs) within a time limit of N_t lesson periods (or time slots).

A suitable schedule needs to be found such that no (teacher, class, student) is in more than one place at a time, and no lecture hall is expected to accommodate more than one lesson at a time or more students than its capacity.

Complexity of Timetabling (TTS)

- TTS is a pure constraint satisfaction problem and it is essentially an example of a resource-constrained multiprocessor scheduling problem, which is an NP-complete problem.
- The problem has been approached by means of linear programming (LP) with binary variables plus some heuristics.
- Using the LP representation, for a set as small as 20 teachers, 30 time-intervals, and 10 classes, the problem would be represented by 6000 variables. This makes the problem intractable.
- The overall complexity of timetabling is in the order of $2^{(N_c N_x N_t)^a}$, for some $a \geq 1$.

For a much simpler problem, which is the *class allocation* (no timetabling part included), using LP, the computational load (n_c) scales as follows:

$n_c \propto N_c^3 N_x^2 N_t^3$, where N_c is the number of room categories (i.e. classroom, auditorium, etc.).

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

The timetabling constraints

In a nutshell, timetabling is the assignment of time slots to a set of events, **subject to** constraints on these assignments.

These constraints are divided into three types:

1. Hard.
2. Medium.
3. Soft.

Hard Constraints The most common and important hard constraint is what is sometimes referred to as an *edge* constraint between two events. This reference is due to the similarity of any **simple** timetabling problem with *vertex coloring* in graph theory; which simply states that any given pair of events $\mathbf{e_1}$ and $\mathbf{e_2}$ must not overlap in time. [more on this, shortly]

Examples of hard constraints

- Events $\mathbf{e_1}$ and $\mathbf{e_2}$ can be viewed as two classes taught by the same professor, or one is a class and the other is a recitation or lab of the same class, etc.
- Events $\mathbf{e_1}$ and $\mathbf{e_2}$ are directly connected if their associated classes do not overlap in time.
- A class can not be assigned to a particular room unless *size of the room* $\geq$ *the class enrollment*. The smaller the difference between these two sizes the better the assignment is.
- If teachers are allocated automatically, specific weekly limits are enforced, as well as limits of how many times various types of classes are taught.
- Other constraints are those directly related to students such as:
  - To sign up for a class or section, it must be open.
  - Eligibility criteria for the section must be met, etc.

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

Medium Constraints

These kinds of constraints do not deal with time conflicts but do have a high cost attached to them. Also, this cost is not as high as that associated with the hard constraints. In the final (legal) schedule the cost of these constraints is minimized and it is quite preferable to reduce it to 0.

Examples of medium constraints

- For each student, balance MWF and $TuTh$ sections.
- Balance or spread out the lectures over the week.
- Classes may request contiguous time slots.
- Classes may be grouped with other classes.
- The overall schedule should have as few “holes” as possible; that is lessons should be glued together, etc.

Soft Constraints

Do not deal with time conflicts and have a low cost associated with them. Also, in the final schedule, this cost is minimized but does not often approach zero.

Examples of soft constraints

- Balance enrollment in multi-section courses.
- Lunch and other break times may be specified.
- Teachers may request periods in which their classes are not taught.
- Do not enroll athletes in courses that conflict with their sport practice time, etc.

Mapping of Timetabling Onto a Graph Coloring Problem

Graph Coloring:

Briefly, given a graph $G = (V, E)$ we have to find a coloring of its vertices with as few colors as possible. The main constraint is that two vertices linked by an edge should not receive the same color (such coloring are called feasible coloring). One could define the set S of feasible solutions as the set of feasible coloring which do not use more than a given number of colors; for example, this upper bound may be the number of colors used in the initial solution.

The Mapping:

1. The courses are the vertices of a graph.
2. The time periods correspond to the colors.
3. Two vertices are linked by an edge if the corresponding courses cannot be given at the same time.
4. Additional constraints have to be taken into account and mapped onto the graph.

Interpretation of the Solution Space

After setting up or mapping the problem we need to determine the following:

- A feasible solution is defined as a schedule that satisfies a subset $(C^{'})$ of constraints (C) : $C^{'} \subseteq C$; provided that all hard constraints are part of $C^{'}$.
- A neighboring solution is obtained by modifying the current solution space (i.e. swapping two or more classes in time).
- Here are two guidelines in determining whether a given constraint to be included in $C^{'}$ or not:
    1. A class should never be assigned to a time period if such an assignment induces only class schedules that do not satisfy all the constraints (specifically the hard constraints).
    2. A conflict between two or more classes should not be a sufficient condition for hindering an assignment.
    3. If all courses last one time period, this is equivalent to changing the color of a vertex, for example.

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

The overall cost function

The *cost function* measures the infeasibility of the current sloution and may invlove the weighted sum of penalties associated with different types of feasibility violation.

We consider the following as part of the function:

- Distance cost: proportional to the distance between where the class is assigned and the building which is housing its home department or school.
- Time period cost consisting of : [**1**] class cost, [**2**] professor cost, [**3**] cost of class spreading over the week and within each of the weekdays, [**4**] cost of class glueing and filling in any holes of the timetable, [**5**] room size cost, and [**6**] students cost which is the sum of various costs associated with each student, such as eligibility criteria and number of classes taken by the student.

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

The overall cost function - Continue ...

- The cost of professor is computed from: **[1]** cost incurred from assigning more classes than the maximum number taught by that professor, plus **[2]** cost from not having a long enough time interval between courses taught by the same professor, plus **[3]** cost of assigning to the same professor two or more classes that overlap in time.
- The cost of room size is computed in a non-linear fashion depending on the magnitude of the difference between the class enrollment and the room size.

Note: *Using summation is not strictly necessary, provided that the components can be combined in some fashion to reflect the quality of the solution.*

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

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 $s$.
```

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

Physical System - Optimization Problem: An Analogy

Physical System	Optimization Problem
System State	Feasible Solution
Change of State	Neighbouring Solution
Energy	Cost
Ground State	Optimal Solution
Rapid Quenching	Local Search
Careful Annealing	Simulated Annealing
Temperature	Control Parameter

The Annealing Schedules

Geometric Annealing

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

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

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.

Reheating As a Function of Cost Some background

- In combinatorial optimization problems, **phase transitions** can be observed as sub-parts of the problem are resolved. Our observations are 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.
- If the best solution found to date is poor (i.e. has a high energy or 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.

The Schedule

The following is due to Abramson et al. and consists of:

- Let C_i to denote the cost of configuration i, and $k = 1$ (the Boltzmann factor).
- At T, the acceptance prob is: $Pr_i(T) = \frac{e^{\frac{-C_i}{kT}}}{\sum_j e^{\frac{-C_j}{kT}}}$.
- For a set of configurations $\mathcal{N}$, the expected cost at T: $< C(T) >= \sum_{i \in \mathcal{N}} C_i Pr_i(T)$.
- The expected square cost at T: $< C^2(T) >= \sum_{i \in \mathcal{N}} C_i^2 Pr_i(T)$.
- The variance of the cost at temperature T is: $\sigma^2(T) =< C^2(T) > - < C(T) >^2$.
- Now, the maximum specific heat is: $T_{msp} = \frac{\sigma^2(T)}{T^2}$.
- $T_{new} = P \cdot C_b(s) + T_{msp}$;

  where T_{new} is the reset temperature, P is a constant, and $C_b(s)$ is the current best cost of the given configuration s.

-bb-error =

Specific heat (T_{msp}) vs. temperature (T).

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

Mapping of Timetabling into Annealing Algorithm

The most obvious mapping of timetabling (TTS) involves the following:

Let max(P) = maximum number of classes assigned to any professor.

1. a **state** is a timetable containing the following sets:
    - P: a set of professors.
    - C: a set of classes.
    - S: a set of students.
    - R: a set of classrooms.
    - $\mathcal{I}$: a set of time periods.
2. a **cost** or "energy" E(P, C, S, R, $\mathcal{I}$) such that:
    - E(P): is the cost of assigning more than max(P) classes to the same professor, plus scheduling one or more courses that causes a conflict in the professor's schedule.
    - E(C): is the cost of scheduling certain classes within the same time period in violation of the exclusion constraint, for example.

- E(S): is the cost of having two or more classes conflict in time; plus cost of having in the schedule one or more classes that really don't meet the student's major, class requested, or course requirements; plus the cost of not having the courses evenly spread over the week, etc.
- E(R): is the cost resulting from assigning room(s) of the wrong size and/or type to a certain class.
- E($\mathcal{I}$): is the cost of having more or less time periods than required, plus cost of an imbalanced class assignments (a certain period will have more classes assigned to than others, etc.).

3. a **swap** (or exchange) of classes : class c_i with class c_j in the set C with respect to time periods $\mathcal{I}_i$ and $\mathcal{I}_j$, and/or with respect to classroom R_i and R_j, respectively.

A **swap** which would either decrease or increase the cost of ΔE is accepted with probability $Pr(\Delta E)$. This probability defines a Markov chain on TTS, and have the distribution in which $Pr(TTS)$ is proportional to $e^{-E(TTS)/T}$. As $T \rightarrow \infty$, the distribution gives the limiting probability of being at a certain state of (P, C, S, R, $\mathcal{I}$).

Using a Rule-Based System as a Preprocessor

The System Components

This system was developed specifically for the problem of academic scheduling. The basic data structures system are:

- **Distance Matrix** of integer unit values between each academic department and every other building under use for scheduling.
- **Course Structure** of each course scheduled anywhere in campus. These structures are capable of linking with each other.
- **Room Structure** of each room (regardless of type) involved in the scheduling process. Like courses, rooms are capable of linking.
- **Time Periods Structure** to keep track of which hour or time slot is occupied and which is not.
- **Department-College Structure** many departments belong to other larger departments or colleges, etc.

0.0.1 The Output of the Rule-Based System

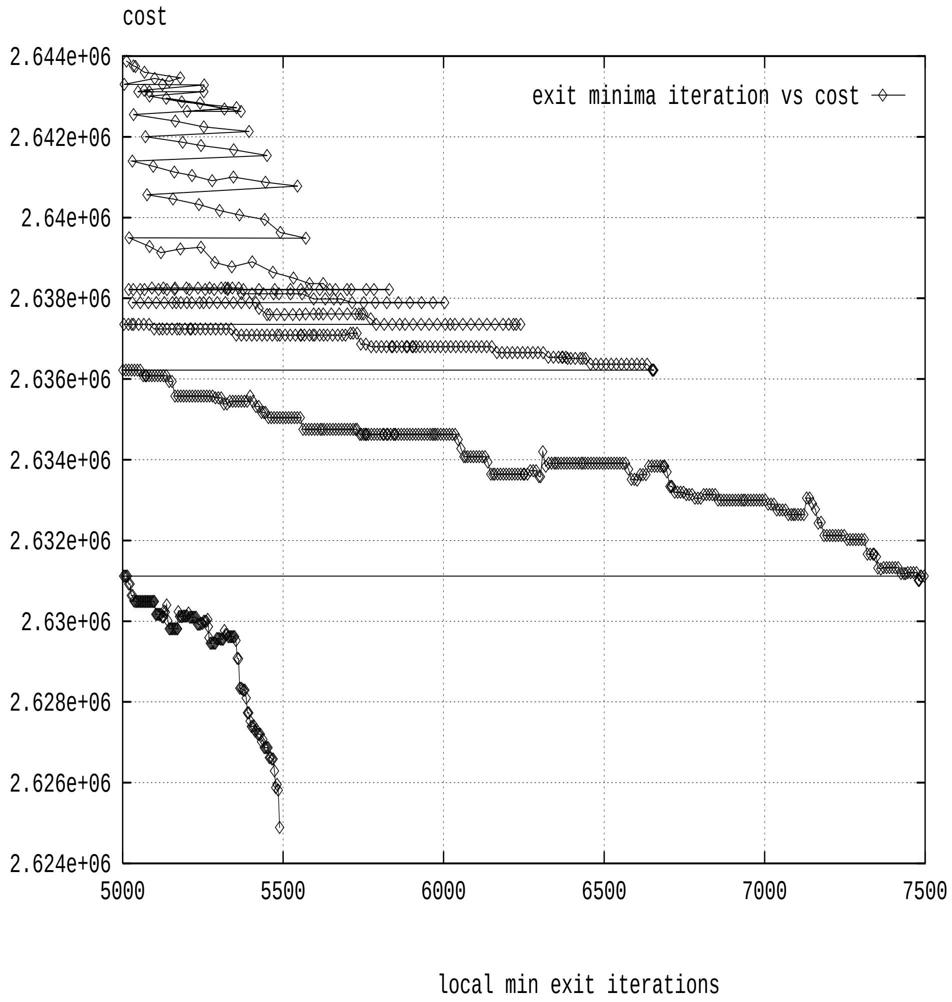
The *first output* of this system consists of classes, with their associated professors and students, assigned to various rooms.

The *second output* is a list of classes which could not be assigned due to time/space or other conflict(s).

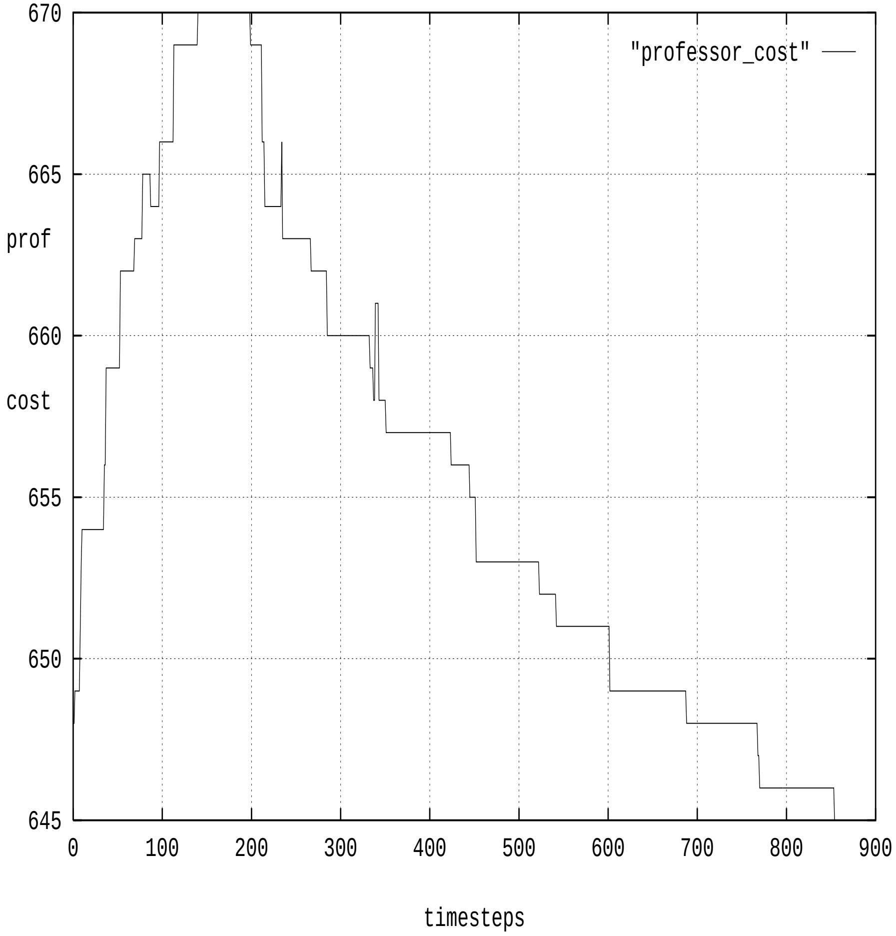
These two outputs are fed to the main heuristic which can be either a *simulated annealing* or a *mean field annealing* to yield a final legal schedule.

0.0.2 The Rationale of Using a Preprocessor

- It is a good practice when using heuristics such as simulated annealing, to start at a good point in the search space or with a partial solution of the problem; regardless of the difficulty of the problem.
- A good starting point can only have a positive effect on the overall solution, and that is improving its quality.
- A good starting point or a partial solution will also have a positive effect on the convergence of the system; faster convergence to the final solution.

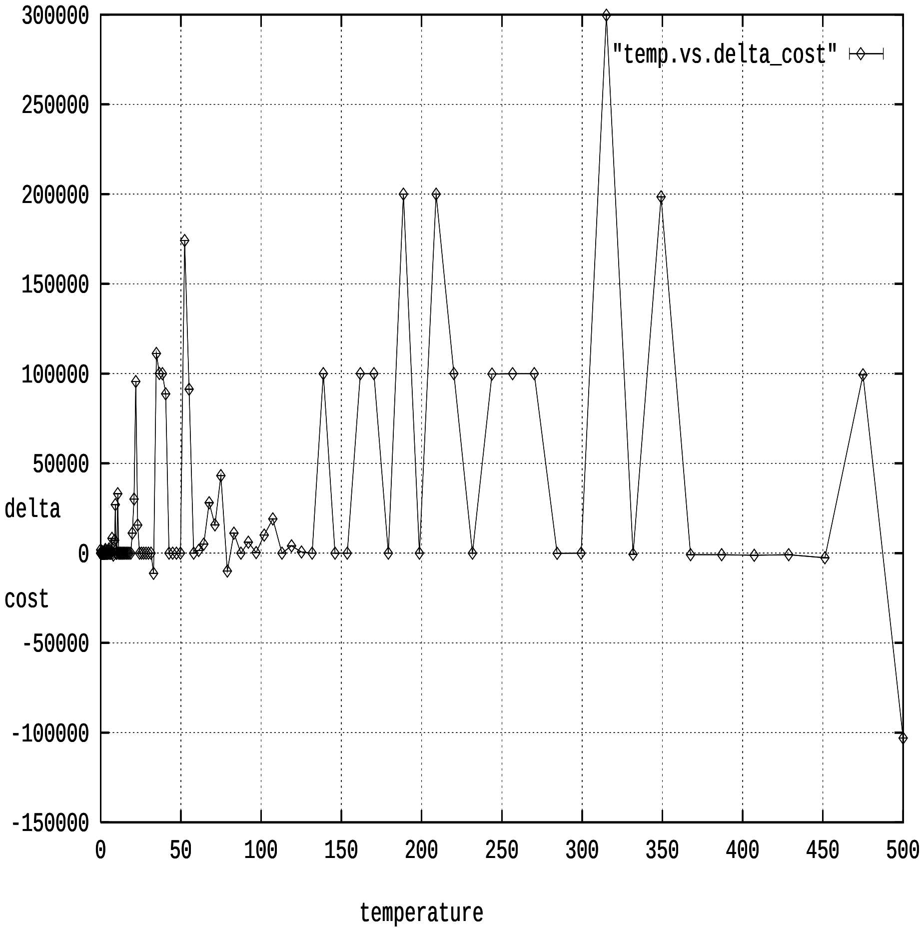


Using reheating based on cost: This figure shows the cost point when local minima is encounterd. A local minima occurs if there is no change at all in cost after a number of iterations.



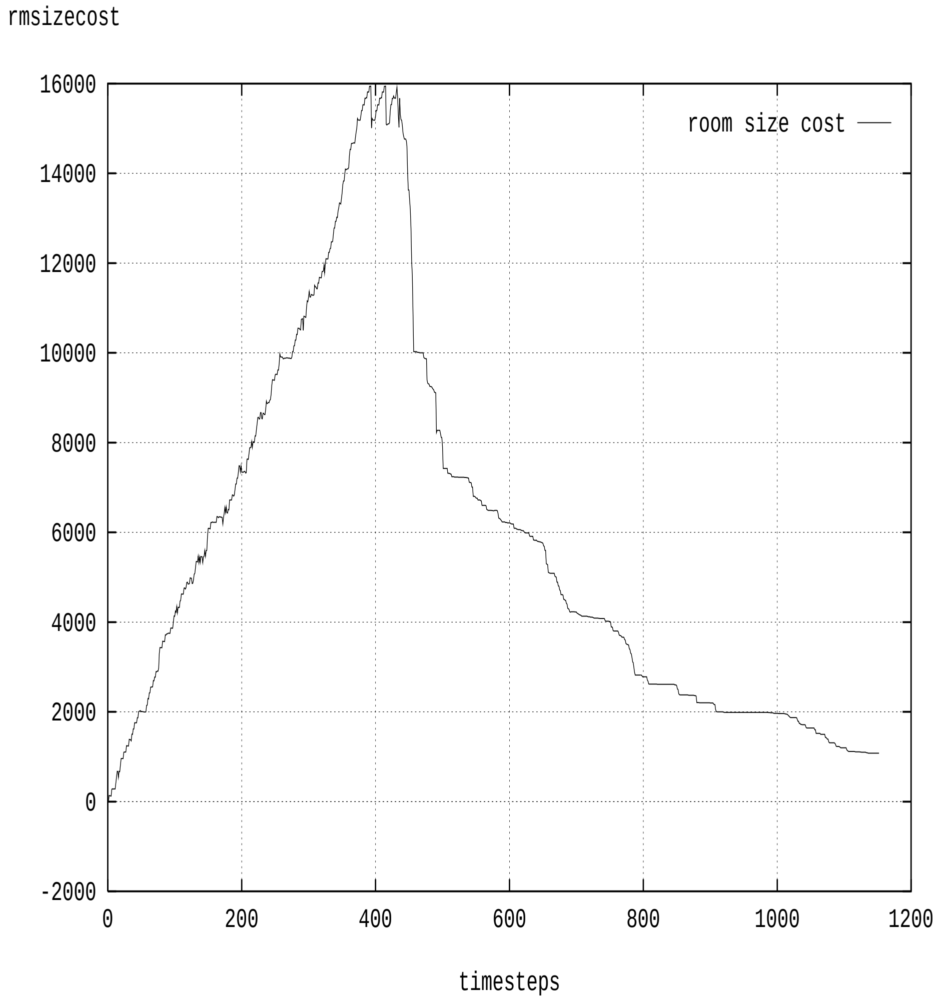
Using reheating based on cost: this shows teachers cost vs. the number of iterations or moves.

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



Using reheating based on cost: This plot shows the change in cost (Δ cost) vs number of iterations.

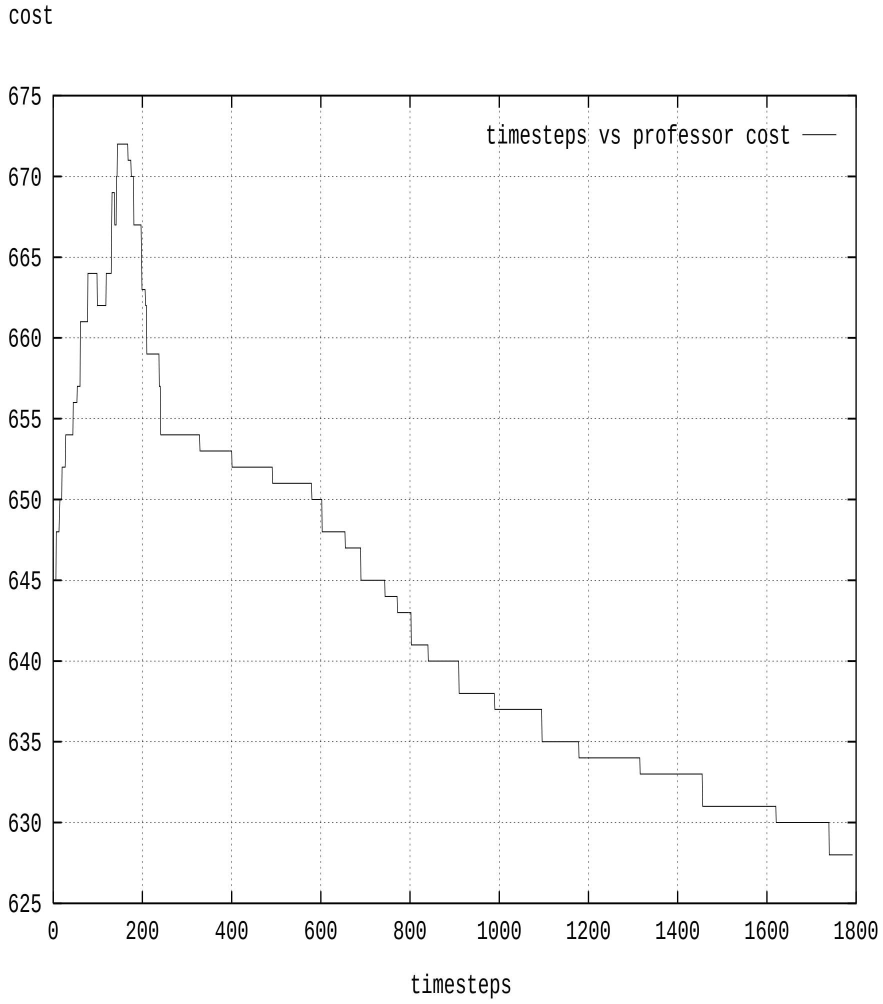
{saleh, gcf, paulc}@npac.syr.edu



Using reheating based on cost: this figure shows the roomsize cost resulting from the given enrollments of classes and their corresponding room sizes.

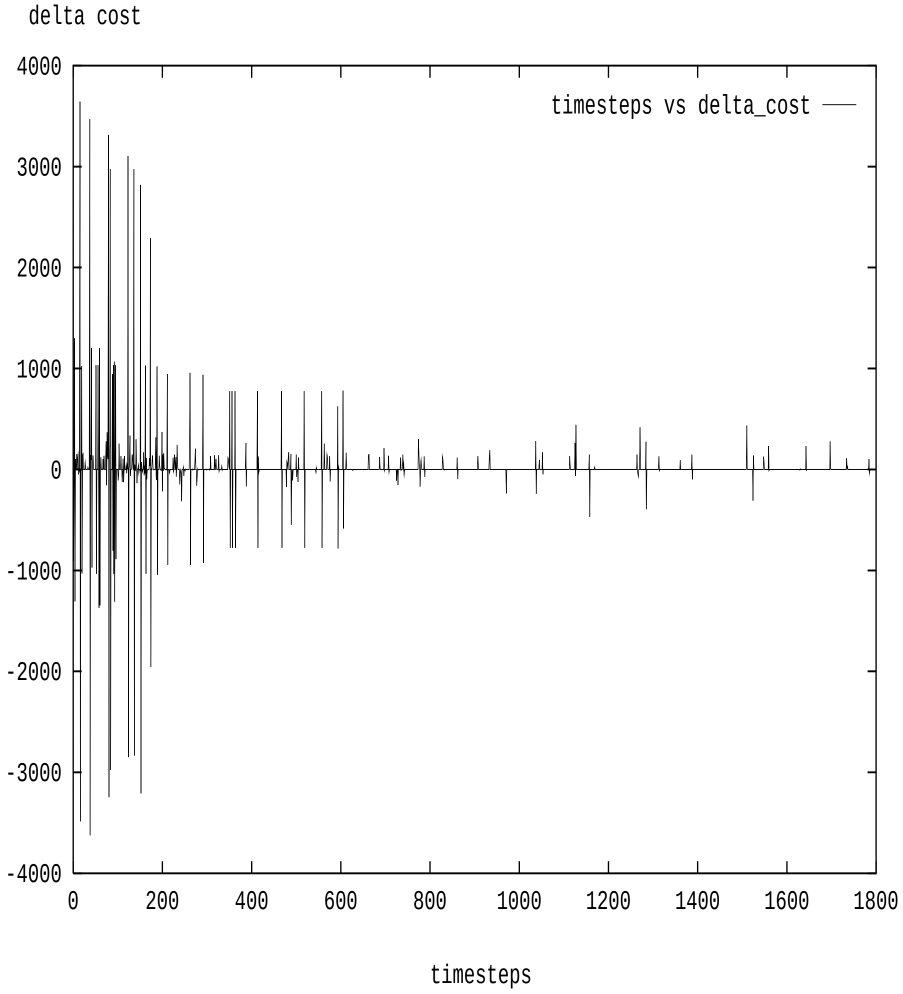
{saleh, gcf, paulc}@npac.syr.edu

Optimization Methods



Using reheating based on cost: this figure shows the cost incurred from assigning professor to classes and classes to professors based on the aforementioned teachers constraints.

Optimization Methods



Using geometric annealing: This plot shows the change in cost while carrying out the simulation. It is timesteps or iterations vs change in the overall cost.

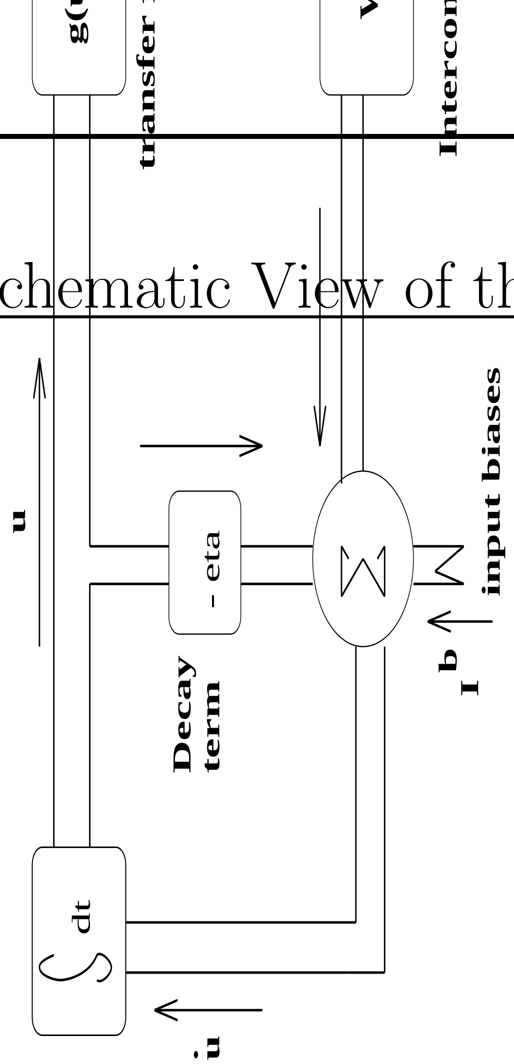
{saleh, gcf, paulc}@npac.syr.edu

THE HOPFIELD NET - A SUMMARY OF THE CONTINUOUS MODEL

- The network is built by connecting a large number of neurons to each other to form a dense, parallel array.
- In general, any i^{th} neuron in the net is described by two variables:
  - its current state (input) u_i, and
  - its output v_i.
- The output is related to the input by a monotonically increasing transfer function: $v_i = g_\lambda(u_i)$.
- The net's dynamics are governed by the first order differential equation: $\frac{\mathbf{du}}{\mathbf{dt}} = -\eta\mathbf{u} + \mathbf{W}\mathbf{v} + \mathbf{I}^{\mathbf{b}}$.
- $g_\lambda(u_i) = \frac{1}{1+e^{-2\lambda u_i}}$, for large λ, this function approximates a step function, forcing v_i to $[0, 1] \implies$ (the output of the net lies on the unit hypercube).
- W_{ij} is the connection strength between neuron i and neuron j.
- I_i^b is an input bias to neuron i.
- Neurons may be updated individually at random (asynchronously) or all together (synchronously).

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

A Schematic View of the Net



The Operation of the Model

- The network's dynamics are governed by a set of nonlinear coupled differential equations of the form:
  $\frac{du_i}{dt} = -u_i + \Sigma_{j=1}^{n} W_{ij} v_j + I_i^b$.
  or the general form of it:
  $\mathbf{\frac{du}{dt}} = -\eta \mathbf{u} + \mathbf{W} \mathbf{v} + \mathbf{I^b}$.
- The I_i^b corresponds to an input bias to neuron i.
- The η term introduces an element of decay into the network's dynamics, which can be used to ensure that **u** variables remain bounded in magnitude.

Update of the Neurons States

To make sure that stability of the of the net is guaranteed, u_i is updated continuously by evaluating $\frac{d}{dt}u_i$ as a function of the total input to the i^{th} neuron.

$$\dot{\mathbf{u}} = -\eta\mathbf{u} + \mathbf{W}\mathbf{v} + \mathbf{I}^{\mathbf{b}} \qquad \ldots\ldots\ldots [1]$$

where

$$v = g(u) \qquad \ldots\ldots\ldots [2]$$

Equations [1] and [2] govern the dynamics of the network.

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

Stability of the Net

Given that **W** is symmetric, Hopfield proved stability for the network by showing that with the above dynamic equations ([1] and [2]) the network output **v** evolves so as to minimize the *Liapunov function*:

$$E^{liap}(v) = \left(-\tfrac{1}{2}\,\Sigma_{ij}\, W_{ij} v_i v_j - \Sigma_i\, I_i^b v_i\right) + \eta \sum_i \int_0^{v_i} g^{-1}(V)\, dV \qquad \ldots\ldots\ldots [3]$$

- $v = (v_1 \ldots v_n)$ is denoting the net state, therefore we have an energy function defined throughout $v \in [0,1]^n$ and dynamics which guarantee convergence to minima.
- For a suitable W, the minima of the first term lie at the hypercube vertices. The second term is minimized at the hypercube center but has negligible imapct at low η.

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

Updating the States of the Net

1. Updating of the states (or the output of the neurons) is carried out either asynchronously or synchronously.
2. From a simulation point of view, we found that synchronous updating a bit more efficient.
3. From a theoretical point of view, it makes not much of a difference on the operation of the net as to which updating mode is used.

Mapping the Problem onto the Network

- A representation of the problem is needed in which the feasible solutions lie at hypercube vertices.
- Each hypercube vertex is regarded as a configuration of the problem, with an associated energy (or cost) function.
- The partition function is then formed by summing the Boltzmann factors of the admissible configurations.
- A quadratic energy function (to be mapped onto the first term of [3]) for which the minima correspond to solutions and the depth of each minimum reflects the solution quality.
- Hopfield and Tank example is the TSP, which they have mapped onto the net using a permutation matrix representation of the TSP tours.

Shortcomings of the Hopfield Model

- The net's partition function sums over a vast number of configurations (e.g. for the TSP, all of the 2^{N^2} configurations are admissible).
- These configurations, again, for the TSP, are nothing like valid tours, and even though these offenders have small Boltzmann weights their large number inevitably affects the thermal average quantities.

Mean Field Annealing (MFA)

0.0.3 Brief Background

To get around the difficulties arising from Hopfield-Tank (HT) algorithm, one way forward is to reduce the computational burden being placed on the network:

- Constrain each neuron representing , for example, a class-professor to be "on" only once (*each class-professor to take place only once during a certain time period.*) by enforcing:

  $\sum\limits_{a} V_{ia} = 1 \;\; \forall i,\, V_{ia} = 1/0 \;\;;$

  saying that event i to take place during interval a or i is the a-th event to occur in the schedule, rather than relying on an energy penalty term to try to do this (as is done in HT).

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

0.0.4 MFA variables: Ising to the Potts model

The Ising Case

The Ising model describes a magnetic system in terms of binary spins $s_i \in \{-1, 1\}$. The system is governed by the energy function

$$E = -\tfrac{J}{2}\sum_i s_i s_{i+1} \quad ,$$

where nearest neighbors interact pairwise with a constant attractive (if $J > 0$) coupling of strength J. The lowest energy state is reached by iterative updating of

$$s_i = sgn[J(s_{i-1} + s_{i+1})] \quad ,$$

which leads to a state where all spins point in one of two possible directions.

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

The Potts Representation

Restricting the state space of the neurons such that exactly *one neuron* at every site is allowed to be on, can be written as:

$$\sum_a s_{ia} = 1 \quad \forall i \quad .$$

Thus for every i, s_{ia} should be 1 for only one value of a, and 0 for the rest.

Choose a k-dimensional vector representation of the neuron states: The a^{th} possible state of the neuron vector $\vec{s}_i$ associated with the node i is represented by the unit vector in the direction of a, so we end up with K different states, $(1, 0, \ldots, 0)$, $(0, 1, \ldots, 0)$, …, $(0, 0, \ldots, 1)$ are all normalized and mutually orthogonal. The number of states available at every node is thereby reduced from 2^K to K. Technically, we now have a K-state Potts model.

Neuron Normalization

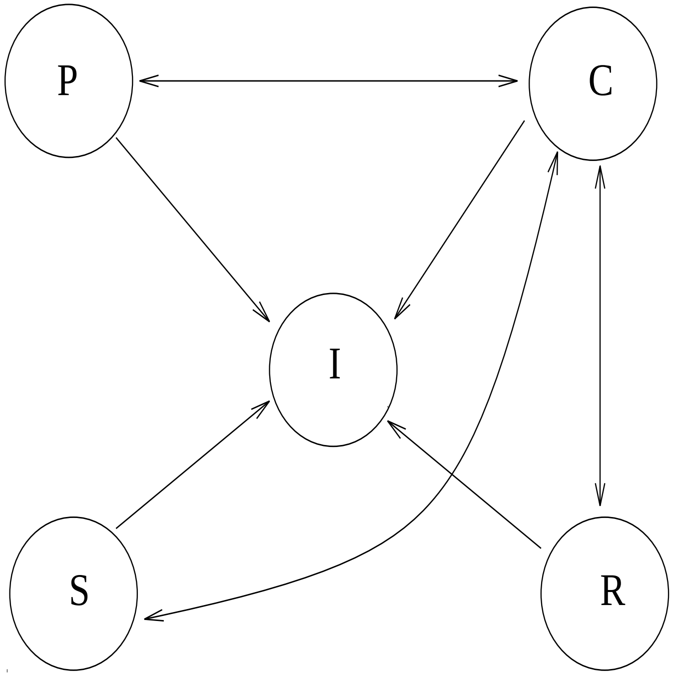
Is the explicit enforcement of

$\sum_{a \in A} V_{ia} = 1$ where A is a set of states, in the MFA algorithm without any need for a penalty term.

0.0.5 Mapping Timetabling onto the Potts Neural Nets

[I]. The relationship between various entities of TTS

Our problem consist of the following sets: teachers (P), classes (C), students (S), rooms (R), and time periods ($\mathcal{I}$).



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

[II]. Neural Mapping

Involve mapping the problem onto the Potts model.

- These multi-state neurons, forming the network, represent different activaties of the timetabling problem .
- Construct the constraints of the problem using these neurons.
- Derive those MF equations controlling the dynamics of the net:
  - Introduce the MF variables , for example, $V_{pq;xt} =< S_{pq;xt} >_T$.
  - Start off with the partition function: $Z = \sum_{[\vec{s}]} e^{-E[\vec{s}]/T}$.
  - Re-write Z in terms of a multidimentional integral over continuous vectors $(\vec{u}_i, \vec{v}_i)$.
  - The MFT approximation to $< \vec{s}_i >$ is given by the value $\vec{u}_i$ at a saddle-point equations:

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

$\frac{\partial E}{\partial v_{ia}} = 0$ and $\frac{\partial E}{\partial u_{ia}} = 0$, generating a set of self-consistency equations in the MF variables:

$$u_{ia} = -\frac{1}{T}\frac{\partial E[\vec{v}]}{\partial v_{ia}} \quad ,$$

$$v_{ia} = \frac{e^{u_{ia}}}{\sum_b e^{u_{ib}}}$$

. Where a denotes the vector components.

For events (pq, xt), we get the following MF variables:

$$u_{pq,xt} = -\frac{1}{T}\frac{\partial E}{\partial v_{pq,xt}} \quad , \quad v_{pq,xt} = \frac{e^{u_{pq,xt}}}{\sum_{x't'} e^{u_{pq,x't'}}} \quad .$$

From the later follows, that $\sum_a v_{ia} = 1$. One can think of the mean field v_{ia} as the probability for the Potts neuron i to be in state a.

- Iterate $v_{pq,xt}$ equation with annealing, i.e. starting at a high temperature, and successively lowering it in the course of the process. A phase transition is passed at $T = T_c$.
- Estimate the critical temperature T_c, which sets the scale of T, by expanding the above $v_{pq,xt}$ equation around the trivial fix-point : at $T = 0$, $v^{(0)}_{pq,xt} = \frac{1}{N_x N_t}$.

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

Definitions

The event-based mapping is carried out as follows (a sample):

- professor(p)-class(c) ⟶ room(r)-timeslot(i), represented by the following neuron:

$$S_{pc;ri} = \begin{cases} 1 & \text{if event } (p, c) \text{ occur in } (r, i) \\ 0 & \text{otherwise} \end{cases}$$

- class(c)-room(r) ⟶ room(r)-timeslot(i),

$$S_{cr;ri} = \begin{cases} 1 & \text{if event } (c, r) \text{ occur in } (r, i) \\ 0 & \text{otherwise} \end{cases}$$

Representation of Some Constraints

We define a number of **hard constraints**:

- An event (p, c) should occupy precisely one space-time slot (x, i). This constraint can be embedded in the neural net in terms of the Potts normalization condition:

  $$\sum_{x,i} S_{pc,xi} = 1 \qquad [P.1]$$

- Different events (p_1, c_1) and (p_2, c_2) should not occupy the same space-time slot (r, i).

  $$E_{X\mathcal{I}} = \frac{1}{2}\sum_{x,i}\sum_{p_1,c_1}\sum_{p_2,c_2} S_{p_1c_1;xi} S_{p_2c_2;xi} = \frac{1}{2}\sum_{x,i}\Big(\sum_{pc} S_{pc;xi}\Big)^2 \qquad [P.2]$$

- A teacher p should have at most one class at a time.

  $$E_{P\mathcal{I}} = \frac{1}{2}\sum_{p,i}\sum_{q_1,x_1}\sum_{q_2,x_2} S_{pq_1;x_1i} S_{pq_2;x_2i} = \frac{1}{2}\sum_{p,t}\Big(\sum_{qx} S_{pq;xi}\Big)^2 \qquad [P.3]$$

- A class q should have at most one teacher at a time.

$$E_{Q\mathcal{I}} = \frac{1}{2}\sum_{q,i}\sum_{p_1,x_1}\sum_{p_2,x_2} S_{p_1q;x_1i} S_{p_2q;x_2i} = \frac{1}{2}\sum_{q,i}\Big(\sum_{px} S_{pq;xi}\Big)^2 \qquad [P.4]$$

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

Relative Clamping

When revising an existing schedule, sometimes it is desirable to have a certain cluster of events to stick together in time (or within some specified period of time). This amounts to making the following notational replacement:

$$i \longrightarrow (j, k) \quad ;$$

where (j) denotes the cluster and (k) an event within (j). In this situation, one has a common t-neuron $(S^{(T)}_{j;t})$ for cluster j but distinct x-neurons $(S^{(X)}_{jk;x})$ for the individual events.

Extended Events in Time To deal with this concept, we use the so called effective *time*-neurons $\tilde{S}^{(T)}_{i;t}$, defined as:

$$\tilde{S}^{(T)}_{i;t} = \Sigma_{k=0}^{g_i-1} S^{(T)}_{i;t-k} \qquad [P.5]$$

where $g_i = 1$ for single hours and $= 2$ for double hours.

Using the above concepts of relative clamping and extended events in time, constraints P.2, P.3, and P.4 becomes:

$$E_{XT} = \frac{1}{2} \Sigma_{x,t} \Sigma_{i \neq i'} S^{(X)}_{i;x} S^{(X)}_{i';x} \tilde{S}^{(T)}_{i;t} \tilde{S}^{(T)}_{i';t} \qquad [P.6]$$

$$E_{PT} = \frac{1}{2} \Sigma_{p,t} \Sigma_{i \neq i'} \delta_{p(i),p(i')} \tilde{S}^{(T)}_{i;t} \tilde{S}^{(T)}_{i';t} \qquad [P.7]$$

$$E_{QT} = \frac{1}{2} \Sigma_{q,t} \Sigma_{i \neq i'} \delta_{q(i),q(i')} \tilde{S}^{(T)}_{i;t} \tilde{S}^{(T)}_{i';t} \qquad [P.8]$$

Spreading of Classes over the Week Classes are spread in a (MWF) and (TuTh) fashion.

- Let the subject attribute be $s(j,k)$. We define an effective "day-neuron" as follows ($\tilde{S}^{(T)}_{j;dh}$ is reduced to a D-type neuron by summing over all hours of the day):
  $$S^{(D)}_{i;d} = \Sigma_h \tilde{S}^{(T)}_{j;dh} = g_j \cdot S^{(T)}_{j;dh} \qquad [P.9]$$
- The penalty term E_{QSD} that spreads classes Q in a particular subject S over different week-days D :
  $$E_{QSD} = \frac{1}{2}\Sigma_{qsd}\left[\Sigma_{jk}\Sigma_h \delta_{q,q(j,k)}\delta_{s,s(j,k)}\tilde{S}^{(T)}_{jk;dh}\right]^2 \qquad [P.10]$$
- Refining the above form by considering 2 different clusters (j) and $(\hat{j})$ of classes over D : (q and s are not used in the summation, since no change occurs to them)
  $$E_{QSD} = \frac{1}{2}\Sigma_d \Sigma_{j\neq\hat{j}} \Sigma_{kh} \Sigma_{\hat{k}\hat{h}} \delta_{q(j,k),q(\hat{j},\hat{k})}\delta_{s(j,k),s(\hat{j},\hat{k})}\hat{S}^{(T)}_{j;dh}\hat{S}^{(T)}_{j;d\hat{h}} \qquad [P.11]$$
- Now for some day (d) of the week:

$$E_{QSD} = \frac{1}{2} \Sigma_{j \neq \hat{j}} \Sigma_{k\hat{k}} \, \delta_{q(j,k),q(\hat{j},\hat{k})} \delta_{s(j,k),s(\hat{j},\hat{k})} \, \Sigma_d \, S^{(D)}_{j,d} S^{(D)}_{\hat{j},d} \qquad [P.12]$$

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

Glueing (or filling in the “holes”)

- The penalty term that rewards situations where (d,h)-events are glued to (d,h-1)-events can be defined as follows:

  $E_{QDH} =$
  $-\alpha \, \Sigma_q \, \Sigma_{d,h} \left[\Sigma_{j,k} \, \delta_{q,q(j,k)} \tilde{S}^{(T)}_{j;dh} \right] \left[\Sigma_{\hat{j} \neq j} \, \Gamma_{q,q(\hat{j},\hat{k})} \tilde{S}^{(T)}_{j;d(h-1)} \right] \qquad [P.13]$

  The energy term can be summarized to say that: *if cluster (j) with events each symbolized by (k) occurs on hour (h) of day (d) then it would be quite desirable to have another event $(\hat{k})$ from a different cluster $(\hat{j})$ to occur on hour $(h-1)$ of that day.*

- Next, since (q) is fixed we could re-write the above equation as follows:

  $E_{QDH} = -\alpha \, \Sigma_{\hat{j} \neq j} \, \Sigma_{k,\hat{k}} \, \Gamma_{q(j,k),q(\hat{j},\hat{k})} \, \Sigma_{d,h>1} \, \tilde{S}^{(T)}_{j;dh} \tilde{S}^{(T)}_{\hat{j},d(h-1)} \qquad [P.14]$

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

Mean Field Dynamics

Let $v^X_{jk;x}$ and $v^T_{j;t}$ corresponding to $s^x_{jk;x}$ and $s^T_{j;t}$, respectively.

Now, substituting v for s in the energy expressions, the MF equations at finite T, are given in terms of the local fields:

1. $u^X_{jk;x} = -\frac{1}{T}\frac{\partial E}{\partial v^X_{jk;x}}$,
2. $u^T_{j;t} = -\frac{1}{T}\frac{\partial E}{\partial v^T_{j;t}}$,

   As the following:
3. $v^X_{jk;x} = \frac{e^{u^X_{jk;x}}}{\sum\limits_{x'} e^{u^X_{jk;x'}}}$.
4. $v^T_{j;t} = \frac{e^{u^T_{j;t}}}{\sum\limits_{t'} e^{u^T_{j;t'}}}$.

The MF dynamics consists of iterating the above equations either *synchronously* or *serially.*

Synchronous Updating: A sweep consists of first computing all *local fields*, and then updating all the neurons; (1)-(2) followed by (3)-(4).

Serial Updating: For each neuron $s^X_{jk;x}$ and/or $s^T_{j;t}$, the *local field* is computed immediately before the corresponding *neuron state* is updated. That means either do (1) then (3) and/or (2) then (4). In our simulation we use this method of updating.

Some Experimental Results

Constraints Dealing with Classes [Type I]

- Those dealing with class capacity/enrollment.
- A professor can not teach more than a class at a time.
- A class should have at most one teacher assigned to it.
- A class and its recitation and/or lab can not be assigned at the same time.
- A limit on the teaching hours per week for a teacher.
- Spreading out classes over the week.
- Filling in the large wholes in the schedule.
- Observing the various distances between buildings.

Constraints Dealing with Students [Type II]

- To sign up for a class or a section, it must be open.
- Eligibility criteria for the section must be met.
- Balance MWF and $TuTh$ sections.
- Observe maximum limit of classes per student.
- Athlets are not enrolled in courses that conflict with their sport pratice time, etc.

Tabulated Results from the Rule-Based System

Using class constraints

	Total Courses	Scheduled	Not Scheduled	% Scheduled
First Term	4400	3168	1232	72.00
Second Term	4082	3189	893	78.12
Third Term	709	631	78	89.00

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

Tabulated Results from the Rule-Based System

Using class and student constraints

	Total Courses	Scheduled	Not Scheduled	% Scheduled
First Term	4400	2992	1408	68.00
Second Term	4082	2881	1201	70.57
Third Term	709	569	140	80.25

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

Using class and student constraints - First Term Data

No Pre-processor Used

	Scheduled	Not Scheduled	% Scheduled
SA (geometric)	2627	1773	59.70
SA (cost-based)	2914	1486	66.22
Mean Field	2151	2249	48.88

NOTE: [1] Annealing is carried out on space AND time. [2] All shown data is averaged over 10 runs.

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

Using class and student constraints - First Term Data

Rule-based system is used a pre-processor

	Scheduled	Not Scheduled	% Scheduled
SA (geometric)	4280	120	97.27
SA (cost-based)	4333	67	98.47
Mean Field	3985	415	90.56

NOTE: [1] Annealing is carried out on space AND time. [2] All shown data is averaged over 10 runs.

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

*

References

[GJ-79] Gary and Johnson, *Computers and Intractability: A guide to the theory of NP-completeness.* Freeman, San Francisco (1979).

[GPS-89] Gislén, Söderberg, Peterson, "Teachers and Classes with Neural Nets", Inter. Jr. of Neural Systems 1, 167 (1989).

[PS-89] Peterson, Söderberg, "A New Method for Mapping Optimization Problems onto Neural Nets", International Jr. of Neural Systems 1, 3 (1989).

[HT-85] J. J. Hopfield and D. W. Tank, "Neural Computation of Decisions in Optimization Problems", Biological Cybernetics 52, 141 (1985).

[GT-86] Gosselin, K. and Truchon, M. "Allocation of Classrooms by Linear Programming", J. Operational Research Society 37, No. 6, 1986, 561-569.

[CHdW-87] M. Chams, A. Hertz and D. de Werra, "Some Experiments with Simulated Annealing for Coloring Graphs", European Jr. of Operations Research 32 (1987), 260-266.

[TD-94] J. Thompson, and K. Dowsland, "Variants of Simulated Annealing for the Examination Timetabling Problem", Working Paper, Statistics and OR Group, European Business Management School, Swansea SA2 8PP, UK. (1994)

[Abramson-94] Abramson, Dang, and Krishnamoorthy, "An Empirical Study of Simulated Annealing Cooling Schedules", Griffith Univ. report, Nathan, Qld, Aus. (1994).

[Lister-93] R. Lister,"Annealing Networks and Fractal Landscapes", Proc. IEEE Int. Conf. on Neural Nets, March 1993, Vol. I, pp 257-262.