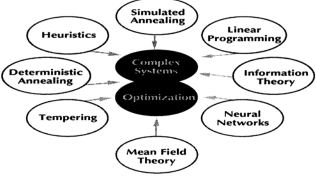
92X Talk

Physical Optimization

Geoffrey C. Fox
Syracuse University
315.443.2163
gcf@nova.npac.syr.edu

6/1/93

Third Keck Symposium on Computational Biology
Houston, Texas - Nov. 1, 1992



- Examples of Physical Computation
  - Cellular Automata
  - Complex Systems
- Physical (Pertaining to Nature) optimization
  - Genetic Algorithms — Evolution
  - Simulated Annealing — Physics(Statistical)
  - Neural Networks — Biology (low level)
  - Information Theory (Maximum Entropy) — Electrical Engineering

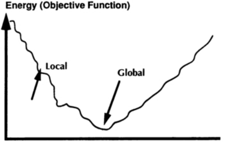
    • • •
  - Elastic Networks — Physics (Determiistic)
  - Deterministic Annealing
- These can be compared with
  - Heuristics — Problem
  - Combinatorial optimization — Mathematics
  - Expert systems — Computer Science (High level reasoning)

- Nature is often solving optimization problems
  - On long term, evolution of species
  - On short term, interpretation of visual and other sensor information
- Physics laws can usually be formulated as variational (optimization) problems
- We can also - and indeed this should be the norm? - combine methods
  - e.g. nature

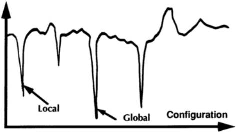
Genetic algorithms	evolve peole over long time period
Expert systems	high level resoning
Learning networks and ?	intermediate level vision
optimization networks	low level vision

Some Questions

- What is the shape of the objective function?
  - Correlated or uncorrelated minima?
- Do you need the
  - exact global minimum
  - an approximate minimum
    - In this case, do you want solution to always be within some tolerance of exact solution
    - or, on the average, to be within tolerance



Local minima in "physics" problems are closely correlated with true minimum



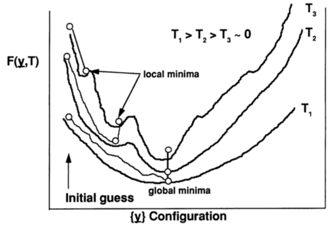
"Computer Science" grand challenges in optimization might look different

- Different methods have different trade offs
  - Robustness, accuracy, speed, suitability for parallelization,
    problem size dependence
- Neural networks do simple things on large data sets and parallelize easily
- Expert systems do complex things on small data sets and parallelize with difficulty
- Combinatorial Optimization
  - Finds exact minima in a time that is exponential in problem size. However in particular cases, e.g. TSP, very clever special techniques make this quite practical - solve exactly $10^4 \rightarrow 10^5$ city problem if we can parallelize
- Physical Optimization
  - Finds approximate minima in a time that is sometimes only linear log(linear) in system size.
  - Sometimes we only want approximate minima

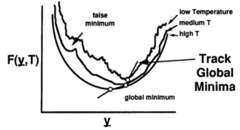
- As computers get more powerful, we need to solve larger problems and physical computation methods get more attractive
  - Thermodynamics studies bulk properties of large systems independent of irrelevant microscopic detail
  - Physical computation can solve 1000 times bigger problems on 1000 times bigger machines
- Illustrates that "computer science" benefits from broad intellectual base that includes biology and physics as well as mathematics and electrical engineering

- **Physical Optimization**
  - **Minimize E = E(parameters $\underline{y}$)**
  - **$\underline{y}$ can be continuous or discrete, or a mix**
  - **Introduce a fake temperature T and set $\beta = 1/T$**
    - **Often T ~ distance scale at which you look at problem**
  - **Probability of state $\underline{y}$ = $\exp(-\beta E)/Z$**

    where $Z = \sum_{\underline{y}} e^{-\beta E}$
  - **As $\beta \to \infty$, minimum E (ground state) dominates**
  - **Find $\underline{y}(T)$ by minimizing $F = E - TS = -{}^{1}/_{\beta} \log Z$**

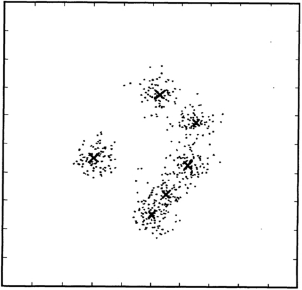


- Annealing tracks global minima by initializing search at temperature T by minima found at temperature T + dT



- Simulated annealing: find y(T) by Monte Carlo as mean over configurations at that temperature
- Neural networks: y is discrete. Find y by mean field approximation
- Elastic net: y is discrete, but use improved mean field including some or all constraints
- Deterministic annealing: leave important y out of sum Σ. Find by simple iterative optimization

The Test Problem (x=real center)



X - center of cluster random generator

A deterministic annealing approach to clustering (Gurewitz and Rose)

- For each data point x there is an energy $E_x(j)$ for its association with the cluster C_j. The probability that x belongs to cluster C_j is:

$$\Pr(x \in C_j) = \frac{e^{-\beta E_x(j)}}{Z_x}$$

- where Z_x is the partition function

$$Z_x = \sum_{k=1}^{c} e^{-\beta E_x(k)}$$

Summing over all assignments of data point x to each cluster

- and F_x is the free energy

$$F_x = -{}^{1}/_{\beta} \log(Z_x)$$

- The total free energy is:

$$F = \sum_x F_x$$

The Squared Distance Energy

$$E_x(j) = |x - y_j|^2$$

$$Pr(x \in C_j) = \frac{e^{-\beta|x-y_j|^2}}{\sum_{k=1}^{c} e^{-\beta|x-y_k|^2}}$$

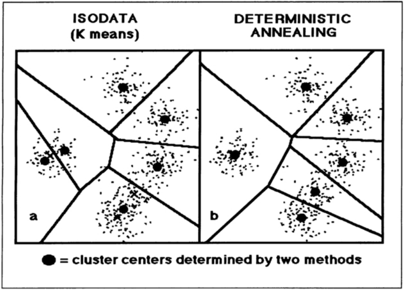
- **Optimizing the free energy:**

$$\frac{\delta}{\delta y_j} F = 0, \quad \forall j$$

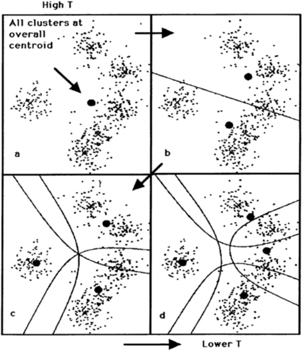
- **yields the solution for the centers of the clusters:**

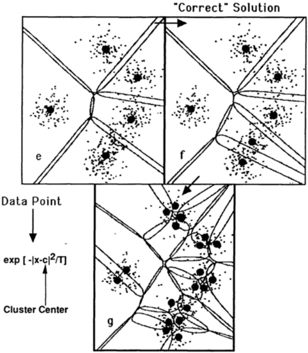
$$y_j = \frac{\Sigma_x x Pr(x \in C_j)}{\Sigma_x Pr(x \in C_j)}, \quad \forall j$$

- **Note $1/b^{1/2}$ is distance scale**

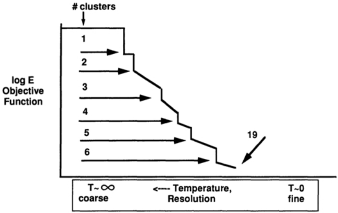

ISODATA
(K means)
DETERMINISTIC
ANNEALING
a
b
= cluster centers determined by two methods

$\exp[-|x - g|^2/T]$ High T is only sensitive to average over large distance scales.





As Temperature $T^{1/2}$ is lowered below cluster size ----> find "clusters" inside true clusters

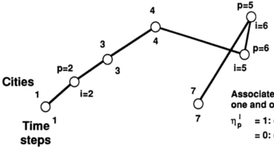


- The clustering problem - like any good physical system - exhibits phase transitions as one lowers the temperature

TSP or Travelling Salesman Problem

Classic NP-complete discrete optimization problem

- **Can be viewed as constrained clustering. This approach "derives" elastic net from deterministic annealing**



Associate each i with one and only one p

η_p^i = 1: city p visited at time i

= 0: otherwise

- **Classical Neural Network Approach to TSP**
  - $\eta^i_p = 1$: city p visited at time i
    $= 0$: otherwise
  - **Constraint**
    for each i only one η^i_p nonzero
- **Elastic net (roughly)**
  - $\eta^i = p$ **multistate neuron**
    (actually position in space of cities)
  - **Simic showed how neural network (Hopfield-Tank) and elastic net came from making different mean field approximations to the same physical free energy**

Generalized Elastic Network
(Simic's derivation of Durbin and Willshaw)

Review of TSP Application

Let p label cities
i label time

TSP: Assign unique i to each p (**)
unique p to each i (*)

constraints

$$
\begin{array}{c|ccccc}
 & 0 & 1 & 0 & 0 & 0 \\
 & 0 & 1 & 1 & 0 & 0 \\
i & 1 & 0 & 0 & 0 & 0 \\
 & 0 & 0 & 1 & 0 & 0 \\
 & 0 & 0 & 0 & 0 & 1 \\
 & 0 & 0 & 0 & 1 & 0 \\
\hline
 & & & p & &
\end{array}
$$

one 1 in each row (*)

one 1 in each column (**)

η **as an N x N matrix**

Such that
$\sum_i d(p_i \to p_{i+1})$
is minimized

goal

Introduce $\eta_p^i = 1$ if salesman at city p at time i, 0 otherwise

- **Typical Hopfield Tank Energy Functions**

$$\underbrace{\sum_{i}\sum_{p,q} d_{p,q}\,\eta^{i}_{p}(\eta^{i+1}_{q} + \eta^{i-1}_{q})}_{\text{goal}} + \sum_{i,p\neq q} \gamma_{pq}\,\eta^{i}_{p}\,\eta^{i}_{q} \underbrace{+ \sum_{p,i\neq j} \varepsilon_{i,j}\,\eta^{i}_{p}\,\eta^{j}_{p}}_{\text{constraints}}$$

time cities

goal

constraints

- **Typical Elastic Net Energy Functions**

$$\frac{1}{2}\sum_{i}|\underline{x}_{i} - \underline{x}_{i+1}|^{2} - \frac{1}{\beta}\sum_{p} \ln \sum_{i} \exp\left[-\frac{\beta\alpha}{2}|\underline{x}_{p} - \underline{x}_{i}|^{2}\right]$$

elastic force between beads

cities beads

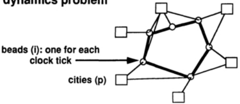
❑ Physical Optimization

- ❑ Energy = Real Goal (Σ d)
  + (*) Positive if constraints violated
  + (**)
- ❑ Consider a physical system with this energy function at temperature T
  Minimize F = free energy = E - TS as T $\rightarrow$ 0
- ❑ Use "mean field" approximation to make annealing deterministic
  e.g. for a term of form η_p^i • function (other η's)

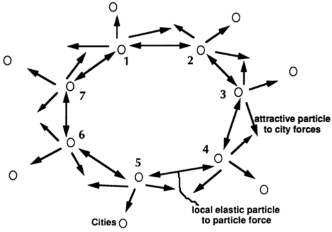
Monte Carlo becomes a deterministic set of equations.
Find minimum of F(mean η's)

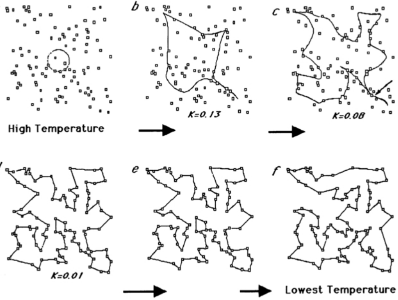
replaced by mean values

- Hopfield-Tank Neural Network
  - Apply physical procedure to full F where we allow η's to vary over all 0,1 values with constraints (*), (**) enforced by penalty functions
- Simic's Derivation of Elastic Net
  - Let η vary over a restricted space where we choose to satisfy some of the constraints exactly.
  - In TSP, one can choose either (*) or (**)
  - Remarkable this converts neural picture into a particle dynamics problem



Physical Model

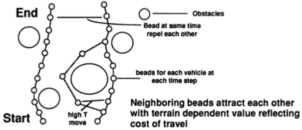



a
b
c
K=0.13
K=0.08
High Temperature
d
e
f
K=0.01
Lowest Temperature

Deterministic Annealing versus Multistate Neurons

- **In elastic net use $\underline{x}$ (i)**
  (position in city space at time i)
- **Equivalently consider a multistate neuron**
  $\boxed{\eta_i \text{ taking values in space } \underline{x}}$
- **Binary neurons ---> Ising model**
- **Multistate neurons ---> Potts model (if discretize $\underline{x}$) or continuous spin model ($\underline{x}$ continuous)**

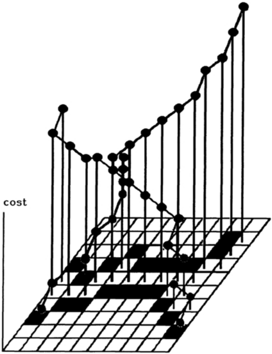
❑ Elastic Net for Navigation



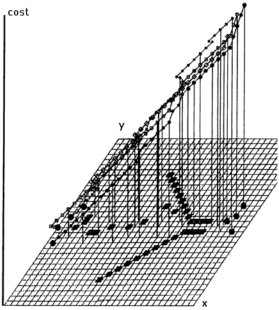
- ❑ Can be applied to single or multiple vehicle problem
- ❑ Note temperature T allows large changes at high T
  e.g. to jump over a large obstacle,
  T is again physical scale
- ❑ Neural Net variables $\eta_i(\underline{x},t)$ are redundant
- ❑ Elastic Net (string) variables $\underline{x}_i(t)$

- New effect: >1 (2 in fact) Lagrange multipliers
- $\exp[-\beta_1 \underset{\text{goal}}{E_1} - \beta_2 \underset{\text{constraint}}{E_2}]$ $\beta_i = \beta_i(T)$
- In this problem, goal (E_1) easy to satisfy but constraints (E_2) hard
- At high temperatures make E_2 go away
  $\beta_2/\beta_1 \to 0$ as $T \to \infty$ implies that $E_1 = 0$
  (elastic string of zero length) is global minimum
- Decrease temperature and gradually "switch on" constraint such that as T-> 0, constraint is rigorously enforced
  $\beta_2/\beta_1 \to \infty$ as $T \to 0$

  e.g. $\beta_1 = 1/T$ $\beta_2 = 1/T^2$

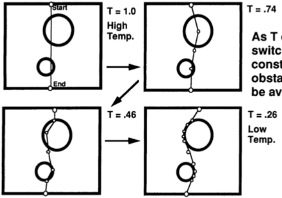


The two-vehicle navigator solution
for a conflict imposing terrain



Four paths in the cost-terrain space calculated by the neural net.

Deterministic Annealing for Navigation (Ghandi, Fox)



As T decreases, switch on constraint that obstacles must be avoided

- Most work on navigation concentrates on exact methods (combinatorial optimization) for a few degrees of freedom
- Physical optimization allows very many vehicles to be navigated
  - Air traffic control
  - Land vehicles
  - Robot manipulators
    Looks very promising for multiple manipulator problem which is otherwise intractable
- But we must use elastic net - neural networks gave us right general idea, but too many constraints

Physical Optimization in Computational Chemistry

Hamiltonian (used) = Hamiltonian (Nature) +

H_c = Hamiltonian (Constraints from experimental measurements or chemical knowledge)

- **Evolve system by Monte Carlo**

  H_c is minimized by Simulated Annealing

- **Evolve system by Newton's Laws**

  H_c is minimized by Deterministic Annealing

Some Applications of Deterministic Annealing

General, Travelling Salesman, Quadratic Assignment
- **Durbin and Willshaw, Frean**
- **Yuille**
- **Simic**

Scheduling
- **Peterson and Soderberg (high school classes)**
- **Johnston (Hubble Space Telescope)**

Track Finding
- **Rose, (Fox, Gurewitz)**
- **Ohlsson, Peterson, Yuille**

Robot Path Planning, Navigation
- **Fox**

Character Recognition
- **Hinton**

Image Analysis
- **Geiger, Girosi**

Clustering, Vector Quantization (coding), Electronic Packaging
- **Rose, et al.**

A new method

Simulated Tempering

Marinari (Rome, Syracuse)
Parisi (Rome)

- Conventional Monte Carlo methods do not work well for Random Field Ising Model = RFIM

$$E = - \sum_{i} \sigma_i \, \sigma_j + \Sigma_i h_i \; \sigma_i$$

nearest neighbor i,j

- Spins σ_i live on 3D lattice
  σ_i = +/-1 to be determined

- Fixed Magnetic Field
  $h_i = |h| \, r_i$
  $|h| = 1$
  r_i = +/- 1 with random probability 1/2

- What sort of NP complete optimization problems are like the RFIM ?
- Normally choose sequence

  $\beta_m = 1/T_m$ increasing

  $m = 0,1,2 \ldots.$

  At each β_m, equiliberate system according to weight function

  $P_m = \exp(-\beta_m E)$
- Monte Carlo gets stuck in local minima (which differ by approximately the square root of the system size from true minima

❑ The Tempering Method

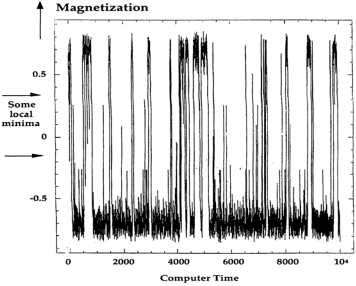
- ❑ β_1 Increasing
- ❑ β_2
- ❑ β_3
- ❑ β_4
- ❑ β_5

Add β to list of dynamical variables
β chosen out of $\{\beta_m\}$
Prob (β, $\{\sigma\}$)
α exp $\{ - \beta_m E (\{\sigma\}) + g_m \}$

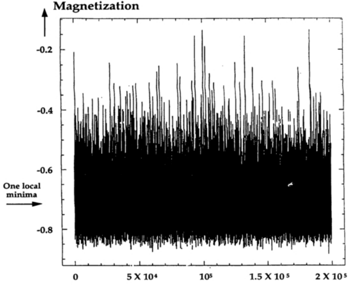
choose so ~equal probability for each m

- ❑ **Now system can move up in temperature and jump barriers**

- ❑ **Don't have to decide on a priori annealing schedule i.e. list of β_m and computer time to be spent at each β_m.**



Simulated Tempering



Fixed β Monte Carlo gets stuck in one particular local minima

Some Scheduling Problems in NASA

- Schedule observations on planetary orbiter subject to physical constraints of power, which instruments near each other, etc. dynamically (new "moon" discovered ---> change schedule). Several hundred people devoted to this at JPL .
- Similar space telescope problem
- Originally NASA hoped to turn shuttle around in ~2 weeks. Actually takes ~4 months
- In orbiter processing facility
    - 60,000 technician hours
    - 10,000 tasks
        - 50% generic
        - 50% mission specific
    - maybe >1 shuttle to share critical teams

University Class Scheduling Problem

Peterson and Söderberg, Lund

- **Currently Syracuse University spends ~3 weeks scheduling freshmen classes. (the time critical case)**
- **Variables are "multi-state" neurons**

  $\underline{\mathbf{x}}_i$

  i = (p,q) = (teacher, class) pair
  $\underline{\mathbf{x}}$ = (classroom, time slot) pair

- **Student (preferences) are the "quanta" of force acting on particles i. These particles move in space of (classroom, time slot).**

Hard Constraints

- $\underline{x}_i$ singled valued i.e. a (teacher, class) event occupies one spacetime slot

- $\boxed{\underline{x}_i \neq \underline{x}_j \quad \text{if } i \neq j}$ *i.e. Different classes do not share class rooms at same time*

- A given teacher can only teach one class at a time

Soft(er) Constraints

- **Each student should be allowed a "lunch class" if possible**
- **The different meetings of a given class should be spread over a week**
- **Each student's MWF and T Th classes should be roughly balanced**
- **Distance constraints between classrooms**

- Student must meet eligibility requirements for class
- Do not enroll athletes in courses that conflict with their sports practice (hard or soft constraints depending on sport)
- Balance enrollment in different sections of a given class
- Balance gender in Honors Seminar sections
- Enroll students in all parts of a course when linked e.g. recitation and lecture
- Satisfy student preferences

Approaches to Complexity

- Dynamical Systems
  - A complicated system is really only sensitive to a few parameters in a space whose dimension can be determined (perhaps not very reliably)
  - Parameters (assumed to be) governed by coupled differential equations ----> chaos
- Statistical Systems
  - Don't ignore the (10^{23}) (other) degrees of freedom but rather sum over them subject to constraints ---> maximize information, entropy ....
- Are approaches consistent?
  - Thermodynamics does show how macroscopic variables emerge from a statistical formulation. How ever only a qualitative relation between dynamical and statistical systems ?
  - "Back-propagation" neural networks unbiased parameterization

Parallel Computing is just an optimization problem, even if can't agree on what to optimize

- ❑ Execution time - main focus of HPCC community?

- ❑ User happiness - main focus of software engineering community

❑ Computing

Problem
↓
Model
↓
Method
↓
High level software
↓
Mix of lever level systems: Fortran, communication systems, ...
↓
Hardware

Minimize:
user preparation time +
execution time
in some combination

- A hierarchy of mapping problems
- We would like to optimize the overall mapping

- Various optimizations are possible at each stage of mapping. These can be (but usually are not) formulated as minimizing

$$E = E_1\,(\text{Performance}) + E_2\,(\text{Constraints})$$

- This is a difficulty if constraint ensures correctness of calculation as in code generation phase of optimizing compiler for a digital computer. Then we must have $E_2=0$ at $T=0$.
- This "correctness" issue might not be present in generation of "programs" for neural (as opposed to digital) supercomputers.
- We can study use of physical optimization in generation or execution of program

Physical Optimization in the Execution of Programs

- Both Message Routing
  Dynamic or static decomposition
  or scheduling of data or processes
  - need "performance" optimized
  - and are satisfied with reasonable (~ within 20% of best) answers
  - and have no difficult correctness constraints

    Heuristics
    Simulated Annealing
    Neural Networks (static data)
    Elastic Nets (messages)

  work well

Find MAP in
DISTRIBUTION (MAP) in FortranD (High Performance Fortran

A user (compiler) generated data decomposition

User (compiler) supplies
- **What is to be distributed**
- **"Graph" - algorithmic connection between entities to be distributed**

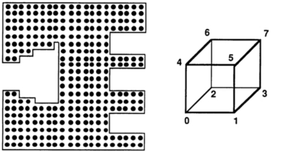
↓

These are:
particles
grid points
matrix elements .... depending on problem

Data Mapping Problem

- **Given: Data set and Algorithm / Solver**
- **Assumptions: Data Parallelism**
  Loosely Synchronous Computation model
  Distributed-memory MIMD multiprocessor
- **Definitions: Computation graph $G_C = (V_C, E_C)$**
  Multiprocessor graph $G_M = (V_M, E_M)$

e.g.



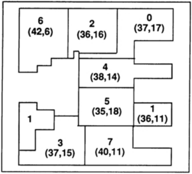
Mapping:

MAP: V_C ----> V_M

such that

<u>***Execution time***</u> of parallel algo/solver is <u>***minimized***</u>

e.g.



Represent mapping configuration by

MAP[v_i] = p_j

Data Allocation Approaches

Allocation problem as a graph isomorphism problem

- a maximum number of pairs of communicating modules fall into pairs of directly connected processors

Allocation problem as a quadratic assignment problem

- $\boxed{\min_{M} \frac{1}{2} \sum_{i=1}^{P} \sum_{j=1}^{P} l_{i,j} \times d(M^{-1}(i), M^{-1}(j))}$

Allocation problem as a minmax problem

- $\boxed{\min_{M} \max_{1 \le i \le P} \{\sum_{j \in N(i)} l_{i,j} \times d(M^{-1}(i), M^{-1}(j)\}}$

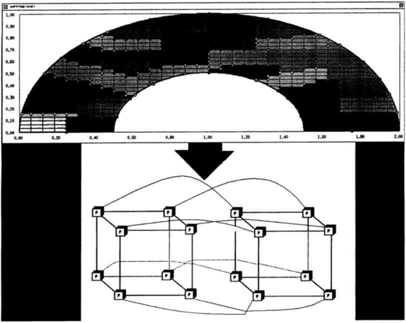
Geometry based allocation approach
- represent the partioning of the domain by a graph
- represent the parallel machine by a graph
- project the problem and system graphs into the 2D-euclidean space
- solve a planar assignment problem
- Iteratively improve the initial assignment

This optimization can also be thought of as finding ground state of a "physics" problem. The particles are "entities" (grid points) moving around in space with "# processors" discrete positions.

Repulsive force - load balance
Attractive force - communicate

Mapping Problem: Criteria

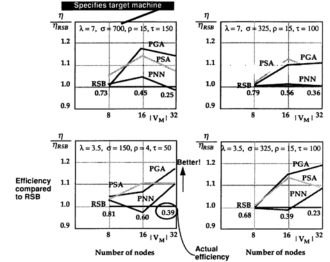
- Decompose the geometric data structures in a specified number of subdomains (substructures) so that:
  - the subdomains have the "same" number of elements or grid points
  - the interfaces amomg the subdomains is "small"
  - the number of adjacent subdomains is minimal
  - each subdomain is compact domain
- Allocate the subdomains to processors, so that:
  - geometrically neighbor subdomains are allocated to neighbor processors in the interconnection network of given parallel machine, and
- Decouple (color) the processors so that:
  - the local synchronization among the processors is edge contention free.



worktoplevel
1.00
0.90
0.80
0.70
0.60
0.50
0.40
0.30
0.20
0.10
0.00
0.00
0.20
0.40
0.60
0.80
1.00
1.20
1.40
1.60
1.80
2.00

4 Typical Cases

N. Mansour (Syracuse PhD 1992)



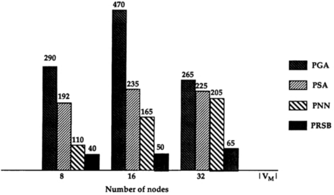
Solution quality for mapping FEMW(2800) with some realistic parameter values

For Machines

RSB = Very good Heuristic for graph partioning

Recursive Spectral Bisection (Simon, NASA Ames)

PSA = Parallel Simulated Annealing
PGA = Parallel Genetic Algorithm
PNN = Parallel Neural Network



Average execution time, in seconds, for mappig FEMW (2800).
($N = |V_M|$)

Actual time taken to perform mapping NOT time taken to execute mapped problem—same size machine used to find mapping as to execute mapped code

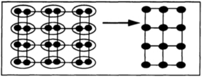
Note:

- Parallel algorithms for
  - Genetic Algorithm
  - Simulated Annealing
  - Neural Networks
  - Are not "trivial"
  - Are not identical to sequential algorithms

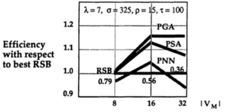
- Large problems (the "real world") require multiscale algorithms just as multigrid is faster than gauss seidel for large P.D.E.'s

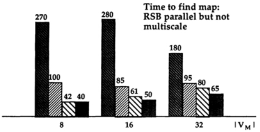
(take $N_{node}^{1/d}$ --> log N_{node}, d=2,3)

Graph Contraction is "computer science" language



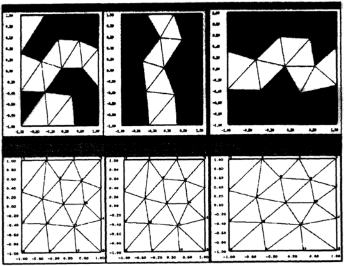
Results for "Multiscale" physical optimization





Any approximate method can be dangerous when applied by

expert - can apply " too precisley"
non-expert - can apply imprecisley



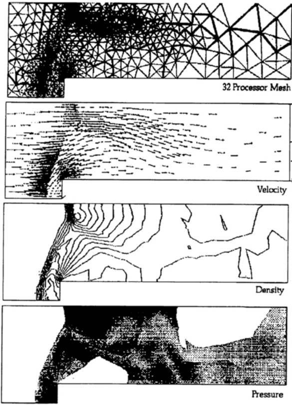
Best known method (Recursive Spectral Bisection) applied to mesh

As given by mesh generator | Sorted in X | Sorted in Y

```
Bob,
Here is my current understanding of the attendance at camp this year

NJ BELL               9 (+3 children) (+2 maybe)
GW                        1
WB                        3 (+1 maybe)
NH                        4 (-1 maybe)
MIT                       1
NY                        2
MICHIGAN              2 (+1visitor +1maybe)
BUFFALO               0 (+1 maybe)
CORNELL               5 (+2 dinner)
              TOTAL   27 +1visitor +3kids (+5 maybe) (+3dinner)
--------------------------------------------------------------

DETAILS

NJ BELL               9 (+ 1 children) (+2 maybe(with 2 kids))
              Bob                     4D
              Nick          3D
              Raju          2D
              Harold        1q
              Carol         1q
              Lalita        1q
              Val                     2q
              Kostas        w
              Michael       w

GW                          1
              Neil          1D ??

WB                          3 (+1 maybe)
              Ken                     5D
              Alan          4D??
              Anna          1D??
              David         ??

NH                          3 (+ 1 maybe)
              Roger         4D
              Frank         2D
              Tao     ??              1q
              Jim??         3q

MIT                         1
              Jenko         1D

NY                          2
              Miyata        5D
              ??                      W

MICHIGAN              2 (+1visitor +1maybe)
              Gary          2D
              Shigako       1D
              <visitor>     visitor

BUFFALO               0 (+1 maybe)
              Joe                     3D??

CORNELL               5 (+3 dinner)
              Todd          3D
              Satoshi       3D
              Iva                     1Q
              Chris K.      1Q
              Neil Choi     1D+
                            Paula    (dinner?)
                     Advisor and spouse


We look heavy in the Dan levels... and looks sort of small in attendance.


Gassho,
Todd
(315)446-8035   (h)     (number was wrong in my letter)
(315)443-5804   (w)
(607)257-7703   (weekends, (Iva's place in Ithaca))
```

Note: Lesson from 1990 CRPC workshop on TSP at Rice

- **The published exact, simulated annealing (etc.) papers have been greatly improved but**
- **Each new heuristic for TSP shows how their method is better than published results which are not "state of the art" ....**
- **Again many say "simulated annealing" or "genetic algorithms" too slow for data decomposition**
- **Only true for some (naive) implementations**


32 Processor Mesh
Velocity
Density
Pressure

- Neural Networks work well for NP-complete load balancing problem but fail for formally equivalent TSP. *Why?*
- Label $\left|\begin{array}{l}\text{data}\\ \text{processes}\end{array}\right.$ by $i = 1 \ldots M$

  nodes by $p = 1 .. N = 2^d$
- The redundant choice of NM neural (binary) variables

  $\eta(i,p) = 1$ i on node p

  $= 0$ otherwise

  Fails for same reason as for TSP
- The nonredundant choice (NO constraints in E) of $M\log_2 N$ neural variables

  $i \rightarrow P(i)$

  $P(i) = \sum_{k=0}^{d-1} 2^k \eta(i,k)$ succeeds

  $\Rightarrow$ Not all NP-complete problems are created equal

In Summary

- Neural Networks work well for data decomposition as neural variables are natural nonredundant description
- In "analogous" TSP and navigation problems, constraints on redundant neural variable $\Rightarrow$ elastic net (can view as a generalized neural net) better
- Why do all methods work so well for graph partitioning when computer scientists are taught to be terrified by such NP complete problems

❑ Program Preparation / Code Generation

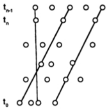
Very Promising

❑ Advice on which algorithms and which machine to use	Expert System
❑ Choose compiler transformations and strategy: loop unrolling, interchange - discrete set of choices at each of many program fragments	Opportunity here? annealing, genetic algorithms?
❑ Given transformation, find performance on given machine	"Back propagation" i.e. learning network
❑ Static Decomposition	Annealing and neural networks both work well
❑ Local Register Assignments, peephole optimizations, pipelining, etc.	Multiple TSP CO/Heuristics now could use elastic nets

as discussed before

Track Finding

- Given a bunch of measurements $x_m(t)$, find the best tracks given certain prejudice as to nature of particles or missiles



- Case I: I -> 20 Tracks
  Kalman Filter or
  c^2 method
  - Do a combinatorial search to choose for each time t_k which measurements belong to which tracks. Use c^2 and various heuristics to reject spurious tracks and accept good tracks

    o Fails if many tracks or lot of noise
    o Difficult to parallelize except on a few nodes

- Case II: Very Many Tracks
  - In one space (plus time) this tracking problem is formally equivalent to edge detection in vision.
  - In vision, measurements are the discontinuities (differentiations) of color / motion / texture etc. in an image. Edge detection involves linking this basic data into "lines" which will separate regions of image.
    - Solve by neural network method

      $\eta(\underline{x})$ = 1 if there is an edge

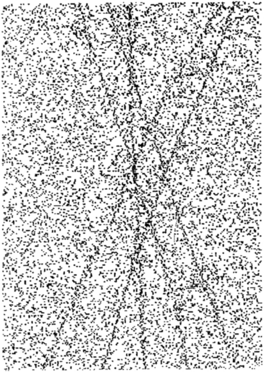
      = 0 if there is no edge

      In vision $\underline{x} = (x,y)$, tracking $\underline{x} = (x,t)$
  - η is a field theory formalism, whereas Kalman Filter is a particle formalism

- **Analogy to Travelling Salesman:**
  - **Hopfield-Tank use $\eta(\underline{x})$ with $\underline{x} = (x,t)$ with t labelling tour and x labelling city**
  - **Two critical differences**
    - **TSP: Only 1 tour so Hopfield-Tank very redundant and must satisfy difficult constraint**
    - **Vision: Many tours and indeed unknown number of tours, so less redundancy and <u>no</u> <u>constraints</u>!**
- **TRW have implemented this approach to tracking**
- **Neural network degrees of freedom independent of number of tracks! (Kalman filter gets difficult as number of tracks becomes large and unknown)**
- **Neural network has essentially unlimited parallelism**

Again neural networks "work" when these are natural degrees of freedom

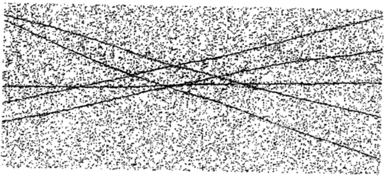
DATA



Gurewitz, Rose

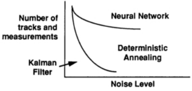
Tracking using Statistical Mechanics

(Deterministic Annealing)



- Case III: Intermediate Number of Tracks or Few Tracks with Dirty Data
  - Now we want to view total tour (track) as a single entity and use either annealing or elastic net methods?
  - Kalman filter fails as it uses information incrementally - adds one time slice to previous local track
  - χ^2 gets stuck in a local minimum - need concept of temperature to avoid false minima
  - Neural networks inefficient as too much redundancy
  - Deterministic Annealing (elastic net) is a good possibility
  - Rose has tested this
  - Note in tracking, tracks are attracted to measurements; in navigation, vehicles are repelled from obstacles, otherwise identical

- So not only do different problems need different methods but in a fixed problem - varying parameter values causes best method to change



- Tracking is like a multiple TSP with each track like a salesman
- The quality of solution needed depends on quality of data. Also as time advances, one can get new information and physical computation naturally allows time dependent data
- E.g. in related navigation problems, you may start with some information about terrain and update it with new sensory data

❑ Conclusions

- ❑ Physical Optimization is a class of naturally parallel heuristics that solve "hard problems" quickly but approximately
- ❑ Monte Carlo or Deterministic
- ❑ Choice of variables is important
- ❑ No universally "good" method even in a given problem, different methods are appropriate for different parameter values
- ❑ Temperature controls a generalized multiscale approach

  Clustering $T^{1/2}$ was distance resolution

  Navigation $1/T$ controlled importance of obstacles
  i.e. T is "resolution" in parameter space

Many Choices - Which is best When?