Algorithmic Chemistry
of future Supercomputers

Gerhard Mack

Seminar Universität Hamburg 2. Mai 1996

Introduction

In 1959, R. Feynman gave a visionary talk describing the possibility of building computers that were "submicroscopic".Recently, L.M. Adleman demonstated the feasibility of carrying out computations at the molecular level by solving a standar NP-hard graph problem (similar to the travelling salesman) using molecules of DNA, and standard protocols and enzymes (*Science*, Nov. 94).

My basic algorithms will run on any *computer with certain minimal capabilities.* Implemented in C^{++}.

Massively parallel architecture

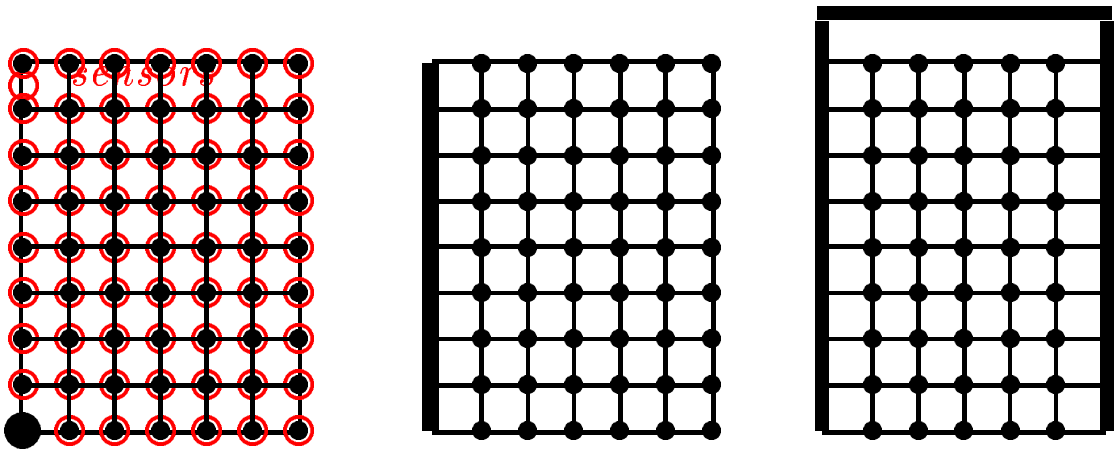


Figure 1: Massively parallel computers

- *Completely homogeneous array of processors with equal basic capacities*
- *Processors only locally connected to neighbours (except possibly for periodic boundary conditions)*
- *One distinguished node for I/O* (**fat**)

Alternative: selfsimilar fractals (snowflake-like) good for multiscale analysis

Anharmonic crystal made of processors:

NEW MATERIAL ("intelligent").

Will be sold by the square foot in some years, hopefully ...

Basic modes of the crystal:

Shock waves in (reasonably few) different "states of polarization".
They can be absorbed and emitted by the processors and thereby interact with each other in a carefully controlled way.

I indicate how to solve basic problems in applied mathematics, such as solution of large systems of linear equations.

Basic types of computations

- Copying
  replication of structure is basic in biology
  "progress" in society is mostly improved copying
- Optimization: *A local cost functional is minimized*

$$f(x) = 0 \iff ||f(x)||^2 = min$$

- Pattern matching, *exact or as good as possible:*
  compilers (parsing), homomorphisms of graphs, find regular expressions, cognition,
  optimal distribution of objects of a system over crystal nodes so that communication is minimized

Formation of shock waves & Updating sweeps

Suppose the passage of a signal through a link leaves the link in a "blocked" state for one time interval so that no signal can pass it backwards.

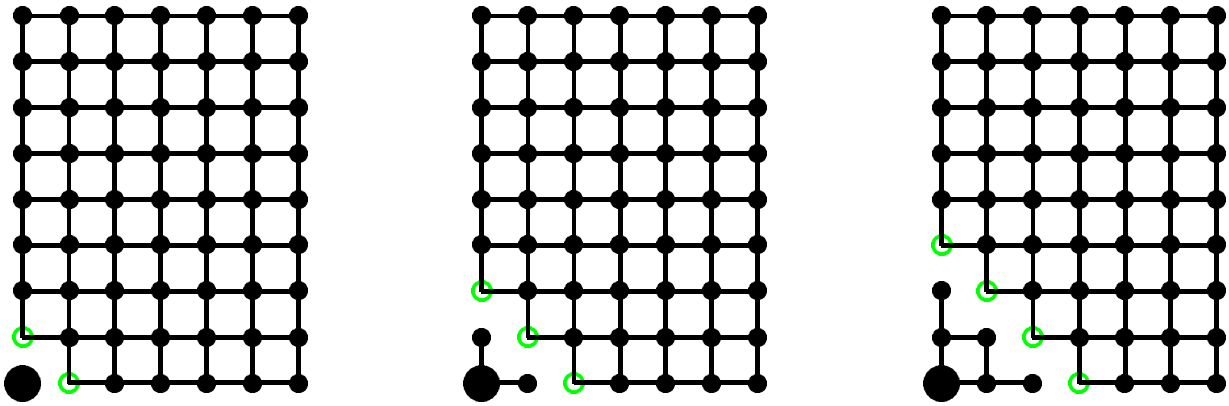


Figure 2: Shock fronts traveling through the crystal

Actually the updating shock fronts will be determines by connectivity properties which come from the problem to be solved, not from the crystal geometry, but optimally, the two should be matched as well as possible.

Collection of results at I/O

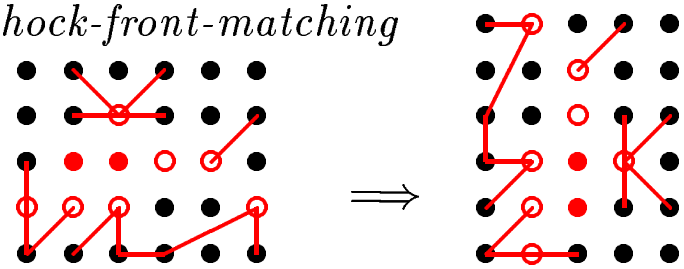
Transport data from the nodes to the I/O node by reflecting the shocks off boundaries or off themselves (e.g. for p.b.c.)


= *first node*

Pattern matching

Example: Operator == for systems (isomorphism)

shock-front-matching



Estimates of Computing capacities

Depends on size, speed and capacity of individual processors

Examples:

1. *20 layers of 10 Mflop processors, each one the size of one pixel, behind a* $640 * 480$ *pixel*^{2} *LCD display*
   $\Longrightarrow$ 300 *Teraflop Laptop*
2. *Individual macromolecules of diameter* 1000 *Å as processors (virus sized)*
   $\Longrightarrow$ *density of processors* $= 10^8$ *times density of neurons in the human brain.*
3. *Presently available:* Java-*chips*, BASIC-*stamps* (1 *cm*^{2}, <100DM)

NB: biochemical processes are slow if *material has to be transported by diffusion.*

Critique of present parallel computing

It lacks general covariance (=coordinate independence) and is therefore a step back into the stone age of computing.

In the stone age, the programmer had to specify the address in memory ("coordinates") of every single variable. Modern programming languages relieve him of this task.

But on present day parallel computers the programmer must again specify the address of the node to which a variable is assigned.

This uses manpower to enhance the performance of the computer. It is a wasteful investment which is worth nothing after few years.

The computer should solve the problem of distributing over nodes by itself by solving an optimization problem, minimizing communication between different nodes.

To formulate this problem properly, one needs a general frame work.

General system theoretic frame work

Thinking - by man or machine - is information processing.
Q: What is the basic type of information? What language to talk about it?

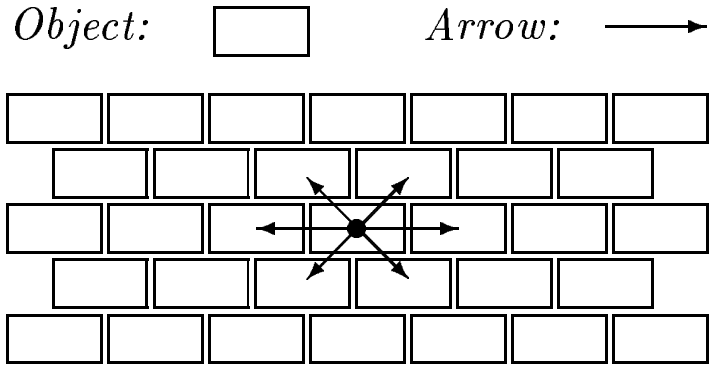
Basic postulates

- *The basic form of information is structure ("topology") all information is coded structure: some special structures are given names ...*
- *A universal meta-language - the language of thought - permits to talk about structure. It is described by a set of axioms which define a system.*

Think of nodes as having internal structure isomorphic to computer as a whole

Mathematical axioms define a language because they specify (among other things) what are meaningful statements.
Hilbert: axioms are implicit definitions

How to describe structure



According to its classical definition, a system is a net of relations between agents.

According to the following precise definition, a system is both a graph and a category.

1. Agents: *Objects X of a category*

   Relations: *Arrows $f : X \mapsto Y$ of a category, so that*

   - *Identity arrow $\iota : X \mapsto X$.* $(\iota \circ f = f = f \circ \iota)$.
   - *Arrows can be composed.*

     $f : X \mapsto Y, g : Y \mapsto Z$ *defines* $g \circ f : X \mapsto Y$

2. Locality: *Certain arrows are singled out as fundamental, all arrows can be composed from fundamental ones. They are the links (of a graph)*

3. Adjoint: *To every arrow* $f : X \mapsto Y$ *there is a uniquely specified arrow* $f^* : Y \mapsto X$ *which can be added to the system if it is not already present.*

4. Ego *One object is singled out*

Composite objects with internal structure: *Systems can be objects of new categories.*

Same description for all scales!

Communication is through Paths:

$$b_n \circ ... \circ b_2 \circ b_1 = C : X_0 \mapsto X_n \ ,$$

different paths may represent the same arrow.

Representation Theorem

Every System can be represented as a communication network

Agents X *: associated state spaces* Ω_X

Arrows f: $X \mapsto Y$ *become maps* $f : \Omega_X \mapsto \Omega_Y$.

Setup like in lattice gauge theory or general relativity, but the state spaces Ω_X *need not be linear spaces and the map needs not be linear.*

From Problems to Systems

Example: Discretized linear partial differential equation

$$\sum_j A_{ij} x_j = f_i \quad \text{sparce matrix } (A_{ij})$$

Objects i, *state* x_i
const. Objects f_i
Links $i \mapsto j$ *if* $A_{ij} \neq 0$, *plus links* $f_i \mapsto i$

$$x_i^{new} = f_i - A_{ii}^{-1} \sum_{j \neq i} A_{ij} x_j \quad \text{Relaxation}$$

All objects linked to object i act on it.

Putting the system on the computer lattice:

Objects $\mapsto$ *nodes, typically many to one*
Links $\mapsto$ *paths on the lattice*
A priori estimate L_{ij} *of communication load of links* ij*, maybe updated at run time. Optimization*

$$\sum_{(i,j)}{}' L_{ij} \cdot pathLength(ij) = min.$$

Sum over links between objects on different nodes.

How to create a world out of nothing

The idea is that all information is purely "topological", i.e resides in the specification of the structure of a system.
Begin with

$0 =$ *system without an object*

$1 =$ *system which contains itself as only object (selfref.)*

Systems are objects!

Now build systems like

$$1011101001 = 1 \mapsto 0 \mapsto 1 \mapsto 1 \mapsto 1 \mapsto 0 \mapsto 1 \mapsto 0 \mapsto 0 \mapsto \mathbf{1}$$

Easy to implement in C++

Universal dynamics

is dynamics which is defined for every system without any need of further (nonstructural) information. It is a rule which determines what structural transformations take place in the course of time, given an initial system.

Example 1: Reproduction fork dynamics

This kind of dynamics is the basis of biological life on earth. It governs DNA replication during cell division.
It is a universal copy machine for **any** *structure*
It is a renormalization group fix point: Same dynamics on all scales

Example 2: Updating fork dynamics

This is used for propagating updating sweeps (shocks).
It is a modifies version of the familiar "breadth first" algorithm for graphs, which "passes without leaving a trace".
Same as reproduction fork dynamics, except the copies of objects are discarded (not made).

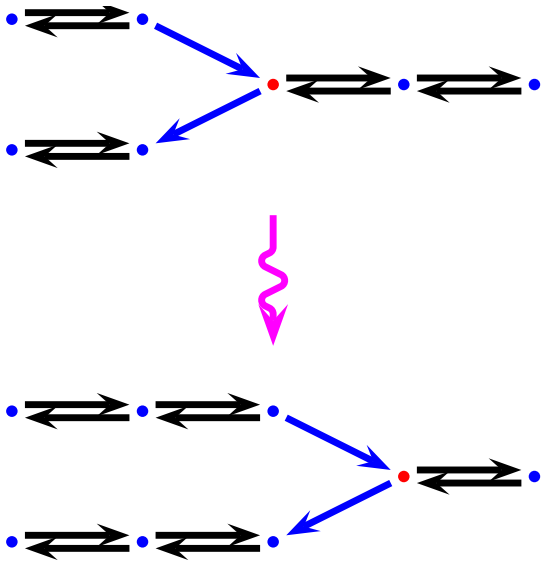
The following moves are defined for every system

1. *removal death of a designated object (or arrow)*
2. *replication of an object (or arrow) (several variants)*
3. *fusion in the absence of frustration (e.g of isomorphic subsystems)*
4. *recovery of missing adjoint*
5. *declaration as fundamental of a composite arrow*

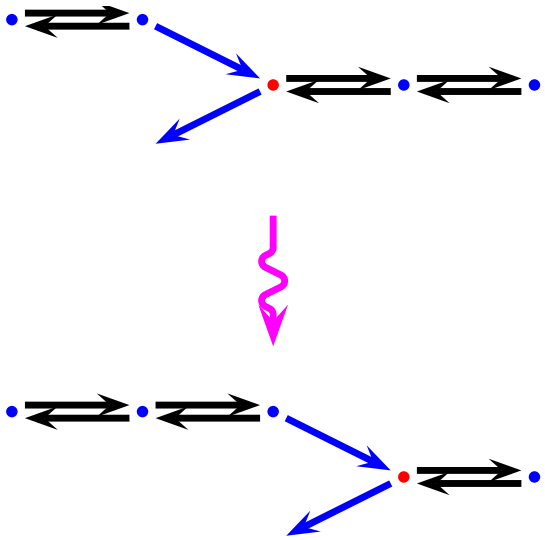
Dynamics fixes which moves take place next, given the present state

5) is the prototype of motion.

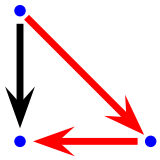
2) Replication fork dynamics:
The red objects are designated to be split in the next move



Updating fork dynamics: similar, copies discarded
This is used for updating shocks



5) A composite arrow becomes fundamental
This is used in local updates



Enzymes

The basic universal local moves form a math. well defined set.

Regard them as Objects called Enzymes which may be parts of systems.

They can act on other objects (radicals=objects with their links). A programme fixes the enzymes linked to node. Shocks can transport or copy these enzymes, or cause the enzymes linked at a node to act.

The enzymes may themselves be modified by the action of enzymes:The program can act on itself.

Also the nodes of the system may procreate (by copy processes) producing e.g, grid refinements in solutions of Maxwell equations:
Subdivide a plaquette (2-dim cell) if the magnetic flux through it exceeds a limit.

Self adaptive computations capability buildt in

Basic capabilities of nodes

Intelligent material consists of molecules called nodes. They must perform basic functions

1. Memory: *store connectivity properties*
   (cp. chemistry: *bonds*)
2. Communication: *dispatch, forward, store messages*
   (cp. chemistry: *vibrations, heat, el. charge, material, light (use firefly enzyme)*)
3. Processing: *binary operations and co-operations*
   (cp. chemistry: *formation and decay of molecules* $A + B \mapsto C \mapsto A' + B'$) *(typically enzyme induced)*

Memory

We may imagine that all information is stored in the form of connectivity properties (structure) of systems.

A system which is constructed from the problem to be solved, must be mapped onto the crystal of processor nodes. Every radical is mapped to some node. A radical consists of an object and a list of links or valences *to it.*

The links are stored in the form of pathways, e.g.

`RUF...` = *first Right, then Up, then Forward*

This also specifies the relative address of the node of a neighbour object.

Internal addresses: Complex nodes may house several objects. They can again be specified by pathways within one node. Imagine that each node has internal structure which makes it a demagnified picture of the whole crystal.

State of an object: Given by the structure of a system again, specified by (relative) adresses of objects and "internal" pathways like `ruf...`*)*

Communication

A message starts with an address header - node address followed by internal

```
UURRRF.rfffu _bla_bla_bla_bla_bla_bla;
```

The node chops off the first capital letter (if there is one) - here U - and forwards the message in the indicated direction - here "up" - to the nextnode. If there is no node-address left, the internal address "`rfffu `*" is used in the same way.*

The address headers of new messages are made from the stored valences.

Redundancy: *Protection against crystal defects (e.g caused by radioactivity in a detector): use also code for addresses of nodes* $A, B, ...$

Adleman: $A = A_1A_2, \quad B = B_1B_2$
location totally unimportant
each half DNA-chain A_i sufficient to identify node.
A_2B_1=Link from A to B. Set of DNA-chains codes graph (connectedness)!

Binary operations and co-operations

A binary operation makes one object out of two. The objects may be composite - i.e. systems.

Example: Binary arithmetic operations, e.g. $+$. Remember this?

$$\begin{array}{r} 4539 \\ +327 \\ \hline 1 \\ \hline ...6 \end{array}$$

This is fusion of two systems into one by a zipper-like local *operation*

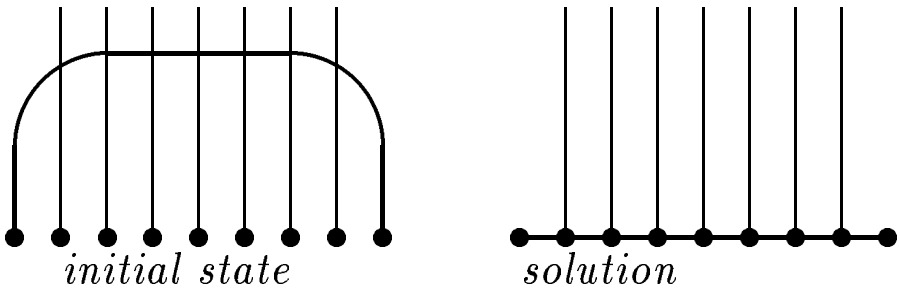
$$\begin{array}{rcl} 4 \to 5 \to 3 \to 9 & \to & \oplus = \\ 3 \to 2 \to 7 & \nearrow & \\ 4 \to 5 \to 3 \to \oplus & \to & 6 \\ 3 \to 2 \to \overset{\uparrow}{\oplus} & \leftarrow & 1 \end{array}$$

Could be coded in DNA: e.g. $0 = AAT$, $1 = ATA$, *find addition-enzyme* $\oplus$
$ATA \oplus AAT = AAT \rightarrow ATA$ *etc.*
NB: 0,1 only structurally different, *same constituents* $\mapsto$ *no* (slow) *transport of material needed in enzymatic reaction (except enzyme itself (ubiquitous)).*

Critical slowing down & Multiscale analysis

Solution of e.g. discretized partial differential equations by local relaxation may suffer from critical slowing down

Example: 1-d Laplace eq. with 0 Dirichlet boundary conditions. $\sum_i |y_i - y_{i+1}|^2 = min$

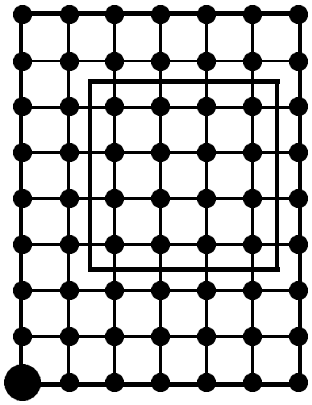


Remedy:

1. *Adapt discretization: coarse the grid locally where possible*

2. *Unigrid version of multigrid method: Global update on extended blocks*

Problem: *Communication to distant sites in same block*
Challenge: *Communication + work* $\propto$ *block volume*
Idea: *shocks confined to block, store gradient of updating proposal across shock fronts*



References

[1] R.P. Feynman, in *Miniaturization*, D.H. Gilbert, Ed. Reinhold, New York 1961) pp 282-296

[2] L.M.Adleman, *Molecular Computation of Solutions to Combinatorial Problems*, Science **266** 1021 (1994)