

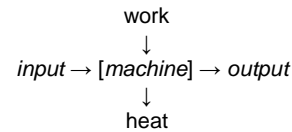
Computing Monadic Fixed-Points in Linear-Time on Doubly-Linked Data Structures

Steven Lindell
Haverford College

LCC 2005

Self-powered automaton

Conventional models are open (and isothermal) systems. Not reasonable as increased miniaturization approaches physical limits of memory capacity and processor speed (in one human generation!). Consider **thermodynamics**:



2

Assumptions

- Individual symbols require a minimum amount of matter, and can be squeezed together to a certain maximum density.
- Individual operations require a minimum amount of energy, and can be executed at a certain maximum rate of power.
- No I/O: computing *in situ* (like Turing machines) on graphs (must pad input for additional work space).

Definition: A machine is *physically scalable* if it can be built and operated using a fixed technology regardless of the total size of the input and the length of the computation.

3

Constraints

Basic premises: $O(1)$ bits/cell $O(1)$ work/step

- In particular, each step is concerned with a fixed amount of information to be processed, and a fixed number of choices for the next move. N.b. Turing was already aware of the asymptotic infeasibility of RAM.
- Assume closed (adiabatic) system. Let size n = mass of the input.
- Propose studying MATTER(n) and ENERGY(n), where *energy is uniformly distributed throughout the matter*.

4

Bounded Degree

Clearly, we should have bounded out-degree, but what about the in-degree d ? Consider entropy loss:



Argument:

1. Information lost upon moving in is $\log_2 d$ bits.
2. This costs kT per bit of energy by the 2nd law.

Spatial arrangement (to **avoid overcrowding**):

- symbols lie in (symmetric) *bounded-degree* data structure

5

Finite Visit

Temporal rule (to **avoid overheating**):

- modify those symbols in a (reversible) *finite-visit* fashion

Argument:

1. because of non-zero transmission losses, any energy beyond a certain radius is inaccessible
 2. within a given radius there is only $O(1)$ energy
- Therefore, ENERGY consumption must be linearly proportional to MATTER available.

6

Data Structures

0	.. contents ..	1
•	.. links ..	•

Uniform: fixed width

Examples: finite collections of nodes (nil pointers have been omitted)

list

Symmetric: doubly-linked

grid

Singular Logic

Variables: $x, y, z \dots$ range over nodes

Constants: $c, d \dots$ are fixed nodes

Properties: $P(x), Q(x) \dots$ to query content bits

Functions: $f(x), g(x) \dots$ to link nodes by pointers

node s

$P(s)$...bits...	$Q(s)$
$f(s)$...pointers...	$g(s)$

7

Logical Complexity

First-order queries

$$\forall x [g(f(x)) = x]$$

Answer: induced graph is (a union of) cycle(s)

i.e. $g = f^{-1}$

Monadic fixed-points

$$R(x) \Leftrightarrow \phi(x; R) \equiv x = r \vee \exists y R(y) \wedge y \sim x$$

where $y \sim x$ abbreviates $adjacency\ x = f(y) \vee x = g(y) \dots$

Idea: Defines a new monadic predicate R recursively

$$\phi(S) = \{x: \phi(x; S)\}$$

Define stages $\phi^0 = \emptyset; \phi^{i+1} = \phi(\phi^i)$ until reaching the least fixed-point $\phi^\infty = \phi(\phi^\infty)$ [nodes reachable from root].

8

Seese's Theorem

Compute first-order sentences in *finite-visit* on bounded-degree graphs.

Hanf's Lemma: For a fixed degree d , any quantifier depth q sentence ϕ is equivalent to a Boolean combination of sentences which say:

$\left. \begin{array}{l} \exists x \text{ says} \\ \text{there are at} \\ \text{least} \\ \text{threshold } t \\ x\text{'s which} \\ \text{satisfy...} \end{array} \right\} (\exists^t x) \tau_r(x)$

$\left. \begin{array}{l} \tau_r(x) \text{ is a local} \\ \text{(quantifier-free)} \\ \text{formula which} \\ \text{depends only on} \\ \text{the neighborhood} \\ \text{of radius } r \text{ about} \\ x. \end{array} \right\}$

Idea of algorithm: For each node x , go out to the radius $r = 2^q$ (constant time because the neighborhood is bounded in size), and determine if $\tau_r(x)$ is true. Count only up to the threshold $t = qd^q + 1$.

9

Locality Lemma

Observation: On graphs of bounded-degree, there are only a fixed number $\tau_1 \dots \tau_N$ of non-isomorphic neighborhoods of radius r .

Define: The *global* (r, t) type is a sentence T_r^t specifying how many radius r neighborhoods of each kind there are up to the threshold t :

Let $\tau(x) \equiv$

be the local r -type of x

(r, t)	1	2	...	t
τ_1	•			
\dots				
τ	•	•	...	•
\dots				
τ_N	•	•	•	•

The table always overflows for sufficiently large structures because it contains only a finite number of dots.

Therefore, there are only finitely many different global (r, t) types.

Lemma (An extension of Hanf's Lemma for sentences to formulas):

Whether an element x satisfies $\phi(x)$ depends only on the global (r', t') type T of the structure together with the local r -type $\tau(x)$.

10

Monotonicity Lemma

Positivity implies ϕ is monotone: $S \subseteq S' \Rightarrow \phi(S) \subseteq \phi(S')$

Naïve algorithm for ϕ^∞ is $O(n^2)$: $S \leftarrow \emptyset \quad S \leftarrow \phi(S)$

$$\emptyset = \phi^0 \subset \dots \phi^j \subset \phi^{j+1} \dots \subset \phi^m = \phi^\infty$$

Note: stages are increasing and stay below fixed-point.

New Idea: don't go stage by stage. Use any ϕ -monotone method.

Lemma: If $S \subset \phi^\infty$ is strictly below the fixed-point, then:

- (a) $\exists s \in \phi(S) \bullet s \notin S$ (ϕ finds something new); and
- (b) $\forall s \in \phi(S) \bullet s \in \phi^\infty$ (everything ϕ finds is safe).

In other words, starting from \emptyset and increasing in the ϕ direction will always reach the fixed-point. In particular, we can add just *one element at each step!*

11

Main Result

On physically feasible (i.e. bounded-degree) data structures, we can compute the fixed-point of a singularly logic formula in linear-time on a *random-access model* of computation (needed to maintain lists).

Proof: mark using DFS. Locality lemma guarantees complexity, allowing us evaluate and update types in $O(1)$. Monotonicity guarantees correctness, allowing us to choose nodes one at a time.

Open Questions

- Under what conditions are singularly fixed-points computable by physically feasible machines (i.e. finite-visit automata)?
- Can result be extended to the full singularly fixed-point logic which includes negation and nested fixed-points?

12