Improving Coalition Performance by Exploiting Phase Transition Behavior

David W. Etherington
Matthew L. Ginsberg
Andrew J. Parkes

Computational Intelligence Research Laboratory

University of Oregon

CIRL

Overview

Focus: **distributed** ANTS computation

Relaxations:

- coping with inconsistencies vs. coping with time-bounds
- interaction with scaling and solution quality

Parallel complexity issues

- O(n) not good enough, need polylog time
- incremental maintenance
- phase transitions
- dependencies, correlation distances

Diagnostic and guidance tools

- Negotiation Graphs/Histories for dependency detection

Relaxations

Relaxations play two roles

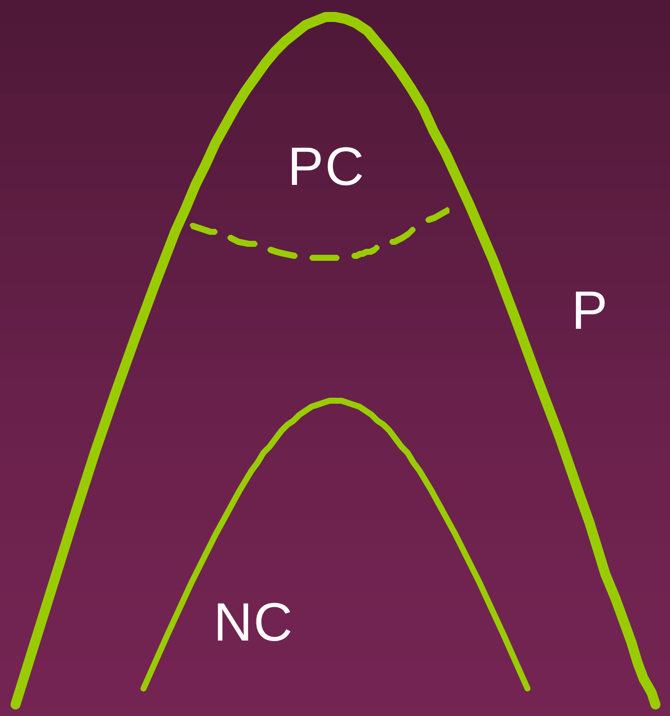
- cope with over-constrained problems
  - drop impossible-to-satisfy constraints
  - use coalitions to encapsulate local inconsistencies?
- cope with time bounds
  - drop difficult-to-satisfy-quickly constraints
  - select relaxation to give good scaling with minimal loss of solution quality; “good enough, soon enough”

P vs. NC

"Good enough" just means "relax to P"?

P (polytime) contains

- PC: P-Complete:
  - believed inherently sequential
- NC: polylog parallel time
  - $O(\log(n)^k)$ time on poly number of processors (exponentially faster)
- Intermediate problems



Relevance:

avoid wasted effort of seeking good parallel solutions to P-Complete problems

Example: SAT

HORNSAT:

$x \,\&\, y \,\&\, w \rightarrow z$

$y \,\&\, z \rightarrow \bot$

$\rightarrow y$

…

Polytime solution,

BUT P-Complete

Believed inherently sequential

and impossible in polylog time

2-SAT:

x or -y

y or z

…

Polytime solution,

AND in NC(2)

Fast parallel, $O(\log(n)^2)$ time, solution

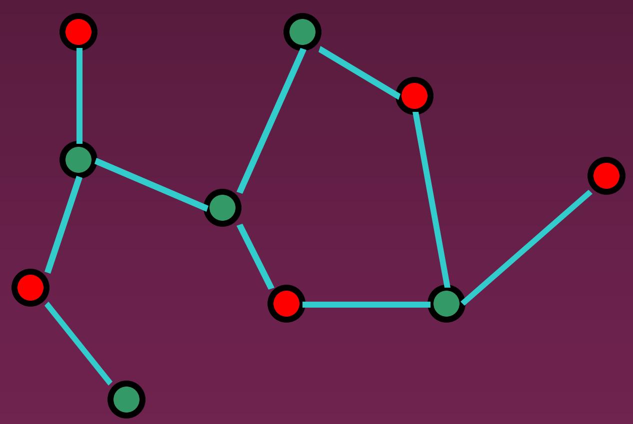
(see Papadimitriou “Computational Complexity” for more details)

CIRL

Example:
Maximal Independent Set

Selected (red) nodes form maxim**<u>al</u>** independent set:

cannot select any more nodes without creating red-red edge



c.f. good sensor coverage without interactions

Spurious Sequentiality

Maxim**al** Indep Set:

cannot add any other node and stay independent.

Linear time sequential: greedily add nodes one-by-one

Can be converted to polylog time [Luby et al]:

parallel *loop: un-committed node probabilistically selects itself*

on pairwise conflicts one node deselects itself

selected nodes commit, their neighbors decommit

Inherent Sequentiality

Maximal independent set (MIS) but also require “lex-first”:

- Pick an ordering of nodes and enforce a preference for nodes that are earlier in the ordering.

Greedy sequential polytime algorithm: pick earliest nodes first.

But is now inherently sequential – problem is known P-complete.

Can guarantee maximality in polylog time but not also lex-first.

Complexity class captures intuition as to difficulty of handling global ordering of nodes/sensors.

Incrementality: Reactive Methods

Sensor problem: targets move, change path.

Dynamic constraint problem.

How fast can we respond to small changes in problem?

- P-complete: believed as hard as initial solution
- NC: still fast
- Intermediate problems: e.g. stable marriage variants Can react in polylog time, even if no polylog method exists for initial solution.

[Sairam, Vitter & Tamassia]

Robust/Efficient Parallelization

Suppose problem takes work W on single processor.

Given p processors hope to solve in time W/p.

- usually unachievable.
- if achieve within a poly(log(n)) factor of W/p then say is
  "efficiently parallelized"
- P-complete problems cannot be efficiently parallelized.

Robustness:

- parallel efficient methods for many different numbers of processors allows greater flexibility in arranging the computation
- with parallel-efficient methods the ANTS have more freedom

CIRL

NC and ANTS

"Soon enough" requires relaxing constraints.

Relaxing to P not sufficient. Need to relax to NC or similar:

P-complete is bad for ANTS.

Even NC is not a perfect match to ANTs:

- NC uses any poly number of processors – this might still be too many – might need a smaller class
- complexity results are worst-case analysis
  - average case or "almost-always" case is adequate? and has a lot better scaling?

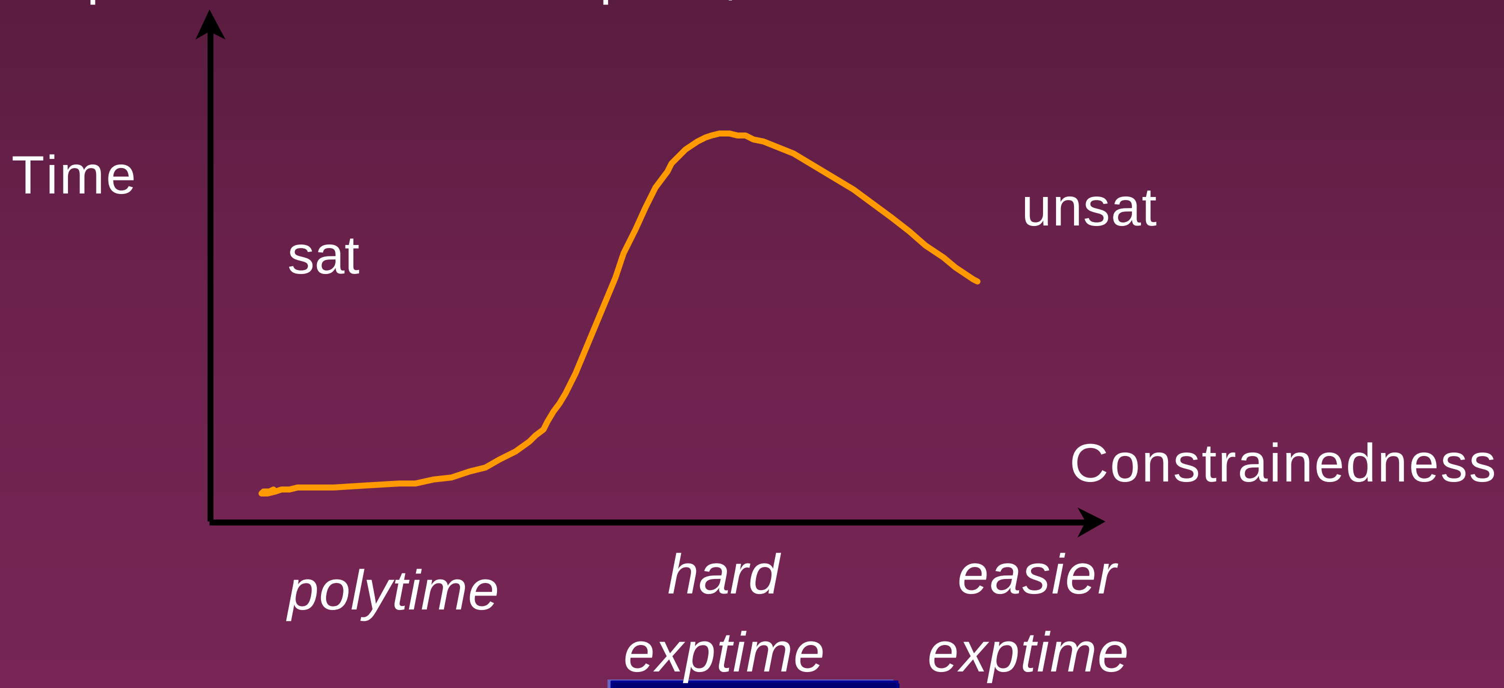
CIRL

Phase Transitions & Polytime

Standard complexity theory considers worst-case.

Average, or even “99% likely,” case can be much better.

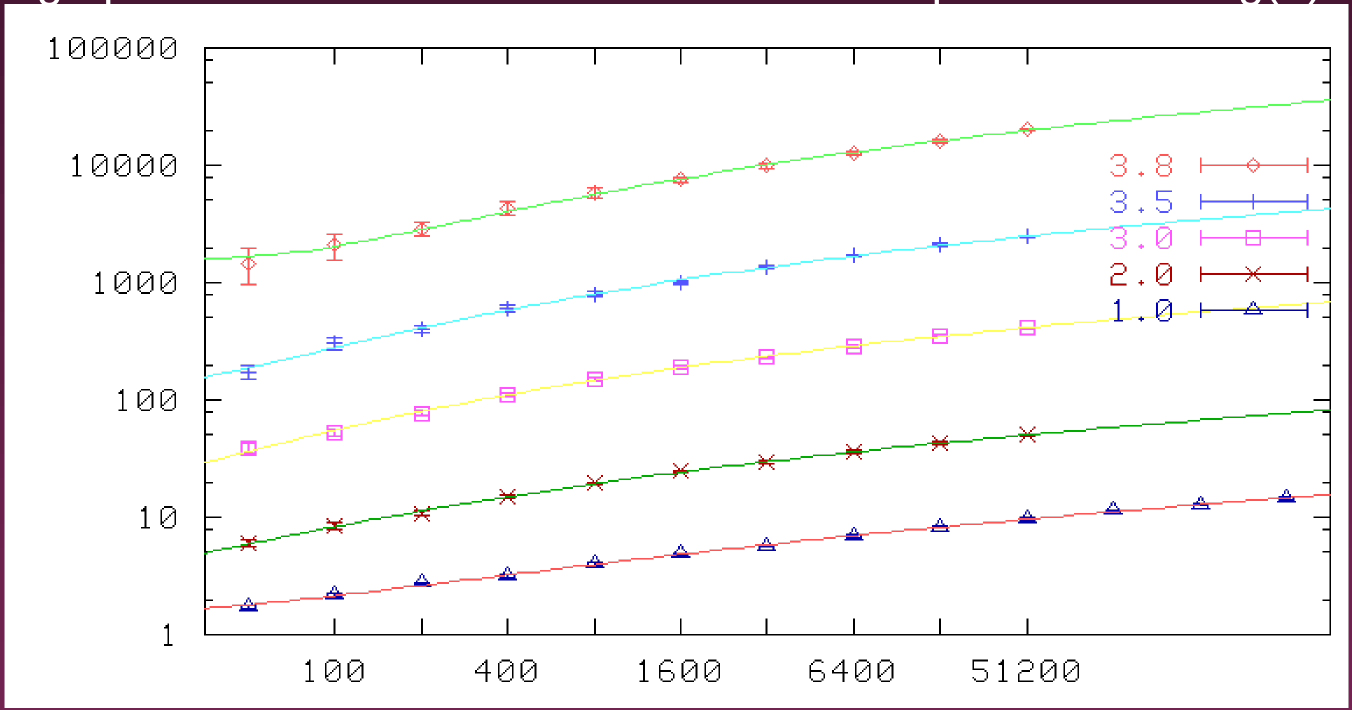
Example: SAT is NP-Complete, but



CIRL

Parallel local search on random 3SAT

Average parallel time   Lines are best-fit quadratics in log(n)



Number of variables, n.

clause/ variable ratios

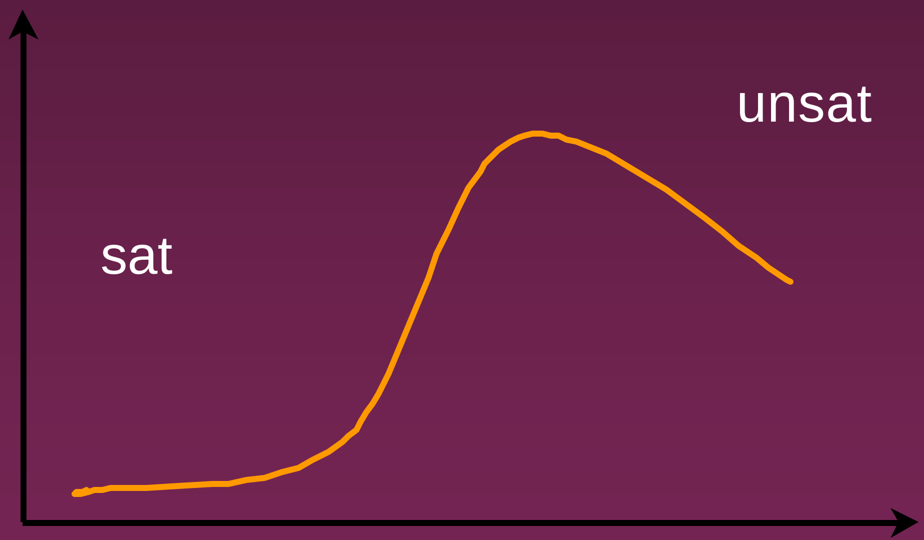
Sequential search is O(n), and have O(n) parallel processors.

Opportunity for exponential speedup is achieved [paper available]

CIRL

Phase Transitions & Polylog Time

Large satisfiable region of 3SAT is solvable in $O(\log(n)^2)$ average time using parallel local search with n processors



Polylog parallel time

Interpretation of PT results

Sharpness of PT means good parallel solutions can be available close to the PT.

In random 3sat the distributed repairs do not fatally interfere with each – despite constraint topology not being simple grid

Expect dependencies between variables to be crucial:

- Fast distributed search is aided by decreased dependencies between sub-problems.
- "If change this variable, which other variables care?"
  - "Just those that are close" – good for parallel properties
  - "Even some that are very distant" – (probably) bad

In physics language: "what are the correlation distances?"

Correlation Distances

Target moves off predicted path: how far does effect propagate?

- All sensors need to co-ordinate with all others? (better not!)
- E.g. cyclic scheduling of sensors activations as graph coloring:
  (optimal) 2-coloring of 2-d grid has infinite distance correlation

Relaxations should prefer to break long-distance correlations

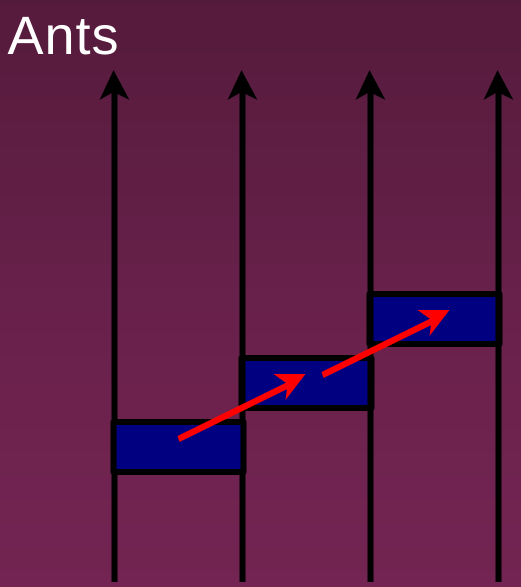
Strong couplings likely to prevent fast distributed search

- How do we detect bad correlations/couplings?
- How to measure whether they are intrinsic or spurious?
- How do we avoid spurious dependencies?

Need diagnosis and guidance tools.

Proposal: analyze negotiation graphs

Negotiation Graphs



Negotiation event

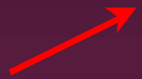
Dependency

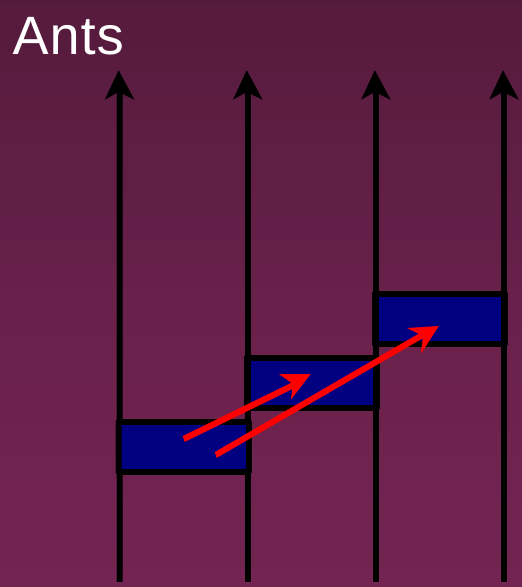
Partime = seqtime = 3

Intrinsic Sequence

Negotiation Graphs


Negotiation event


Dependency



Partime=2, time = 3

More parallelizable

Depth of graph = parallel time

Num Nodes of graph = total work

Shorter dependency chains,

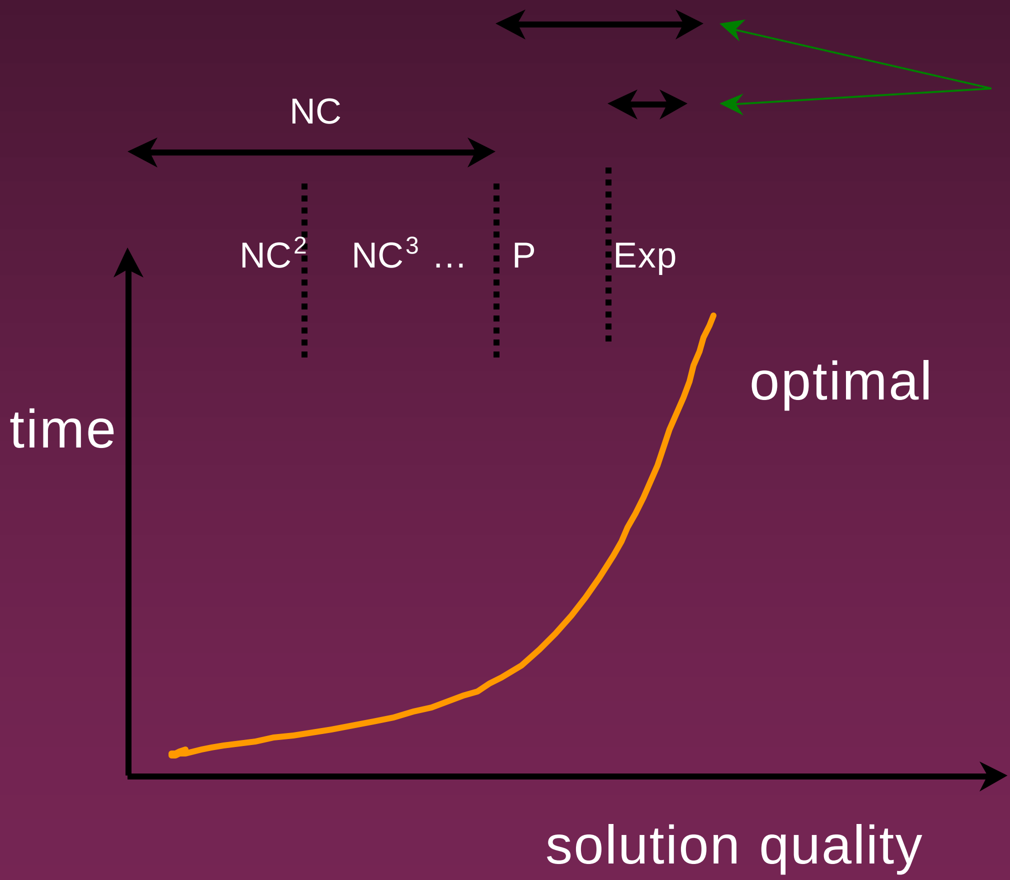
smaller correlation distances?

Other Progress

- Dependency-Maintenance Methods
  - Pseudo-Boolean Constraints: e.g. $x + y + z \geq 2$
  - More compact than SAT, but still general-purpose
  - Implemented Relevance-Bounded Learning: discovers implied constraints during search, stores at most a poly number of them

- Exploiting “Solution Clusters” for Distributed Search [paper available]
  - Coalition of ANTS gives “complex local problem”
  - Exchange complex messages: Solution clusters
    - Exponential number of (candidate) solutions to interface between coalitions
    - Polynomial size representation
  - Aim: Better utilization of communication channels

The Big Picture



Relevant gap corresponds to "cost" of time-bounds

- Determine range of ANTS applicability
- Guide infrastructure management
- Select appropriate negotiation method for current class

Plans. 1

Improve theoretical understanding of relations between

- Relaxations
- Parallel complexity, P/NC
- Incremental computations (dynamic CSPs)
- Phase transitions
- Correlation distances
- Coalitions
- Topology of interaction graph (some cases will be tractable)

Study domains with structure close to that of sensor problem. E.g. grid-like SAT or CSP problems, simplifications of sensor problem to graph-coloring [c.f. Kestrel], etc.

Plans. 2

Interact with ANTS-1 groups to

- Understand the relaxations used in practice:

  quantify what each approach means by “good enough” e.g.

    - target coverage lost?
    - over-usage of power resources?
- Understand the solution methods
  - ♦ inherently sequential components?
  - ♦ sequential-but-fixable?
  - ♦ already parallel?
- Build tools to diagnose bad relaxations and correlations
- Incorporate tools to guide systems

Integration/Transition

Use results of theoretical investigations to guide designs

Diagnosis methods:

- negotiation histories – find dependencies used by search
  (available from groups such as U.Kansas, etc)
- explicit methods to find correlation distance – find true dependencies
  (potentially immediately relevant to graph-coloring studies of Kestrel)

Guidance methods:

- coalition guidance
- detect long-range correlations endangering the parallel properties

Summary

Not all polytime problems can be done in polylog parallel time:

- some are inherently sequential
- distributed ANTS need to avoid relaxations that create such problems

Problems close to phase transition can be effectively polylog parallel time, not merely polytime.

Negotiation graphs/histories:

- extraction of parallel properties and correlations.