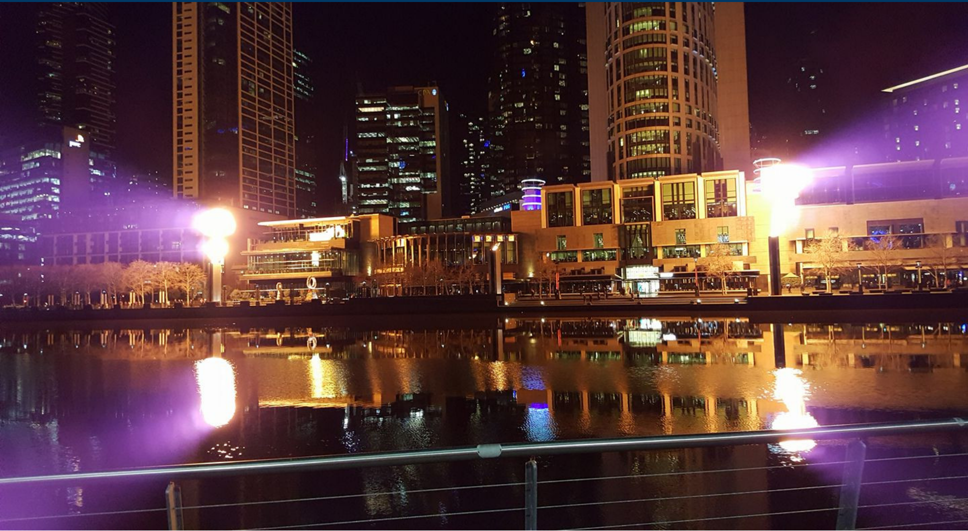
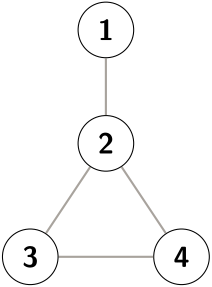
Subgraph Isomorphism in Practice

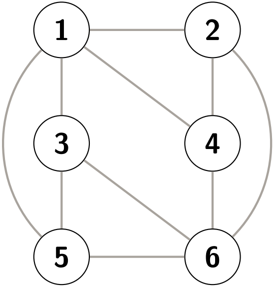
Ciaran McCreesh, Patrick Prosser and James Trimble

University of Glasgow

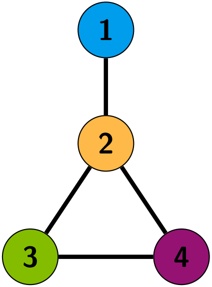


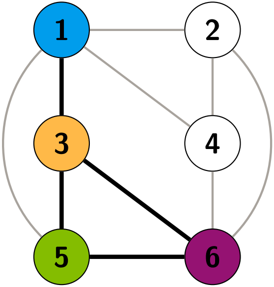
Non-Induced Subgraph Isomorphism





Non-Induced Subgraph Isomorphism





The Algorithm

- Recursively build up a mapping from vertices of the pattern graph to vertices of the target graph.

In Constraint Programming Terms...

- Forward-checking recursive search.
- A variable for every pattern vertex.
- Initially, each domain contains every target vertex.
- After guessed assignments, infeasible values are eliminated from domains.
  - All-different constraint.
  - Adjacency constraints.
- If we get a wipeout, we backtrack.

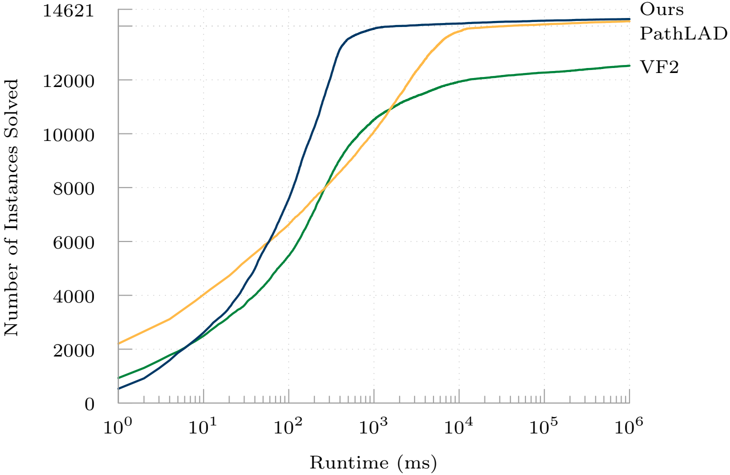
But wait! There's more!

- Clever filtering at the top of search using neighbourhood degree sequences and paths, to reduce the initial values of domains.
- Pre-computed path count constraints, propagated like adjacency constraints during search.
- Bit-parallel implementation.
  - Weaker than the usual all-different propagator, but much faster.

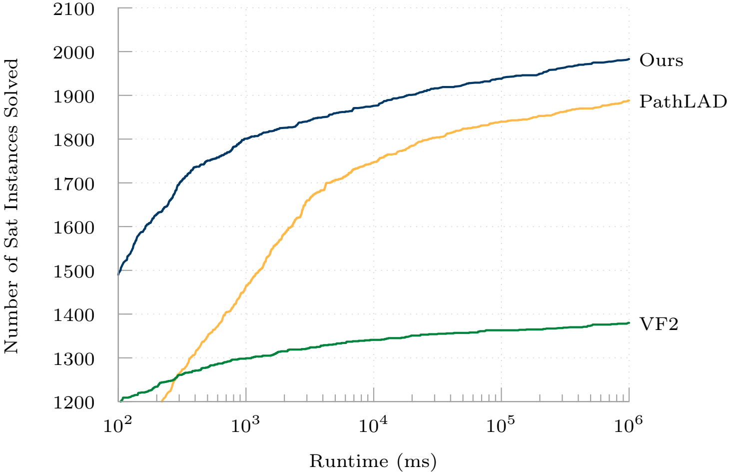
Benchmark Instances

- Other papers: a subset of 5,725 instances from Christine Solnon's selection:
  - Randomly generated with different models.
  - Real-world graphs.
  - Computer vision problems.
  - Biochemistry problems.
  - Phase transition instances.
- Here: all 14,621 instances:
  - $\geq 2,110$ satisfiable.
  - $\geq 12,322$ unsatisfiable.
- A lot of them are very easy for good algorithms.

Is It Any Good?



Is It Any Good?



Search Order

- Variable ordering: smallest domain first, tie-breaking on highest degree.
- Value ordering: highest degree to lowest.
  - The main focus of this talk.

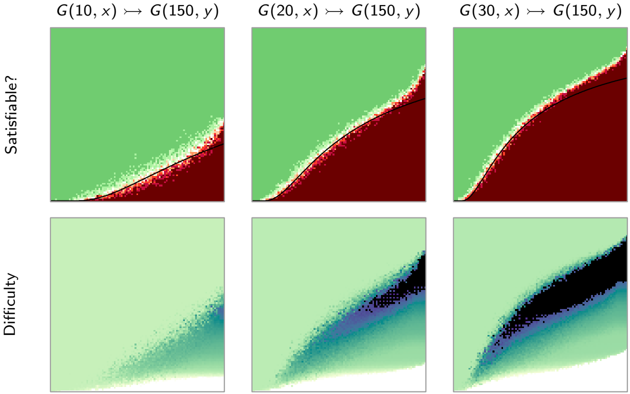
Hand-Wavy Theoretical Justification

- Maximise the expected number of solutions during search?
- If $P = G(p, q)$ and $T = G(t, u)$,

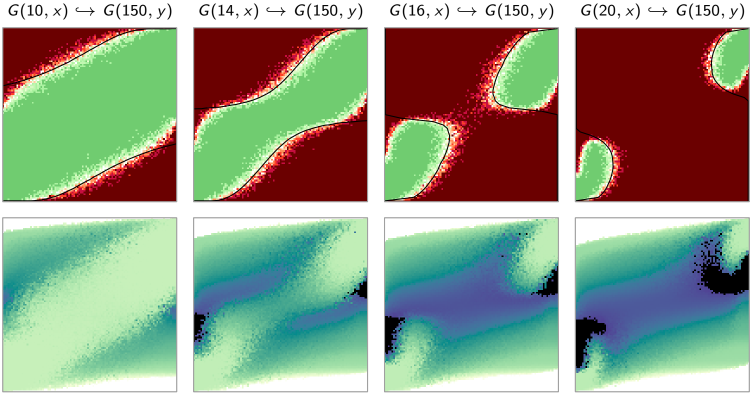
$$\langle Sol \rangle = \underbrace{t \cdot (t-1) \cdot \ldots \cdot (t-p+1)}_{\text{injective mapping}} \cdot \underbrace{u^{q \cdot \binom{p}{2}}}_{\text{adjacency}}$$

- Smallest domain first keeps remaining domain sizes large.
- High pattern degree makes the remaining pattern subgraph sparser, reducing q.
- High target degree leaves as many vertices as possible available for future use, making u larger.

Phase Transitions



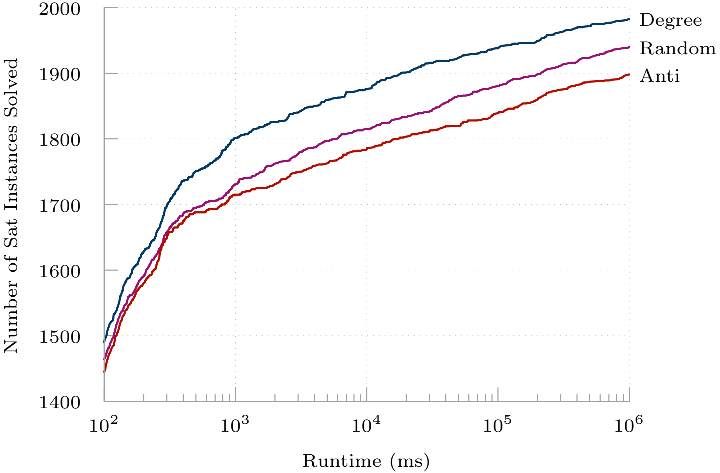
Incidentally, Induced is Much More Complicated



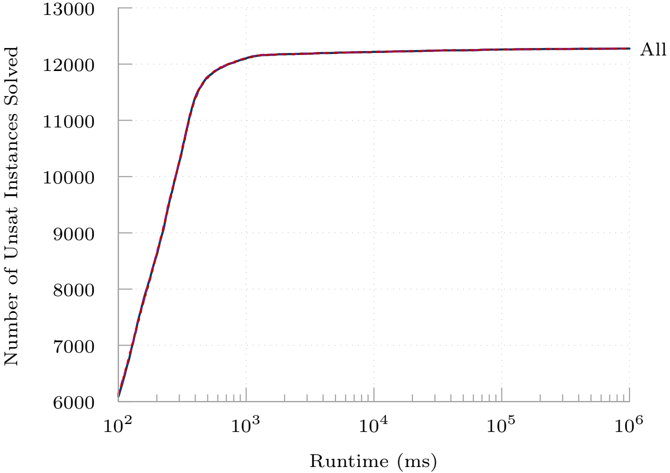
More on Value-Ordering Heuristics

- Highest degree first.
- But degree spread is usually quite low. . .

Sanity Check



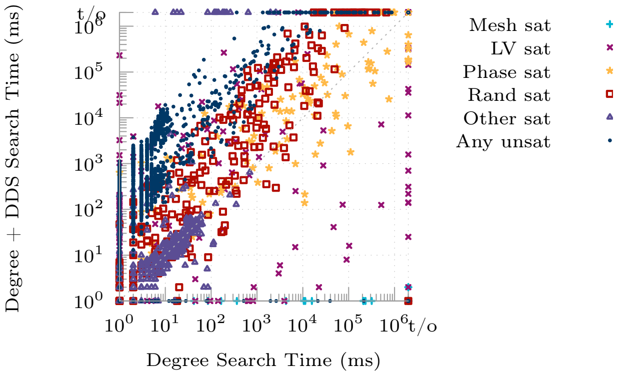
Sanity Check



Discrepancy Search

- Going against a value-ordering heuristic is called a *discrepancy*.
- Conjecture 1: the total number of discrepancies to find a solution is usually quite low.
- Conjecture 2: early value choices are most likely to be incorrect.
- Definite fact: early mistakes are exponentially more expensive than later mistakes.
- Discrepancy searches offset this, at the cost of introducing overheads.

Discrepancy Search



Restarts

- Run search for a bit, and if we don't find a solution, restart.
- Count number of backtracks, restart using the Luby sequence (with a magic constant multiplier).
  - 1, 1, 2, 1, 1, 2, 4, 1, 1, 2, 1, 1, 2, 4, 8, ...
- Obviously, something needs to change when we restart.

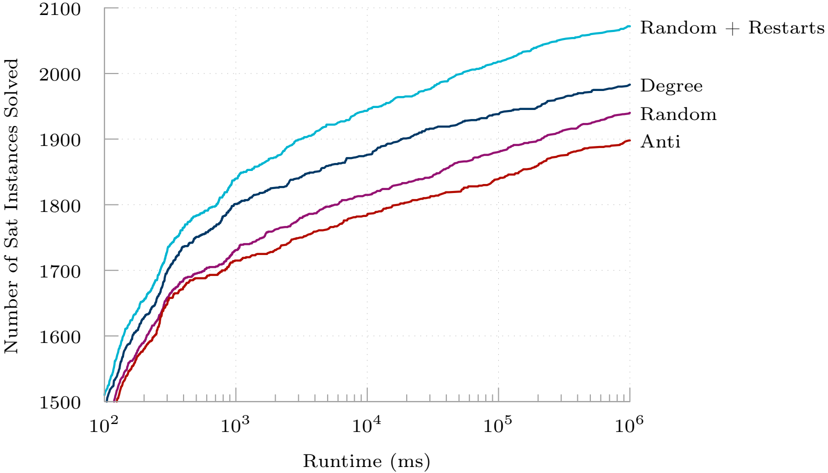
Nogoods

- Whenever we restart, post new constraints eliminating parts of the search space already explored.
- Potentially exponentially many constraints.
- But they are all in the form

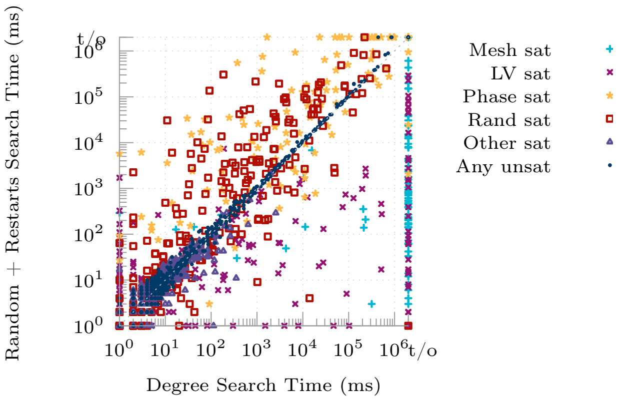
$$(d_1 = v_1) \land (d_2 = v_2) \land \ldots \land (d_n = v_n) \rightarrow \bot.$$

- Use two watched literals to propagate in $O(1)$ish time.
  - Basic idea: clauses only propagate when exactly one $(d_i = v_i)$ literal has not been set to true.
  - Watch *two* literals per clause that have not been set to true.
  - When unit propagating, only look at clauses with a watch corresponding to the assignment made.
  - Either find a new literal to watch, or propagate.

Random Value-Ordering with Restarts



Random Value-Ordering with Restarts



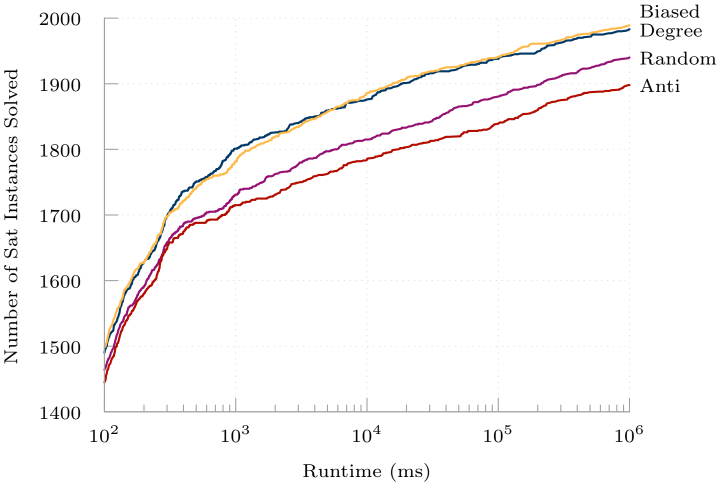
Biased Value-Ordering

- Select a vertex v' from the chosen domain D_v with probability

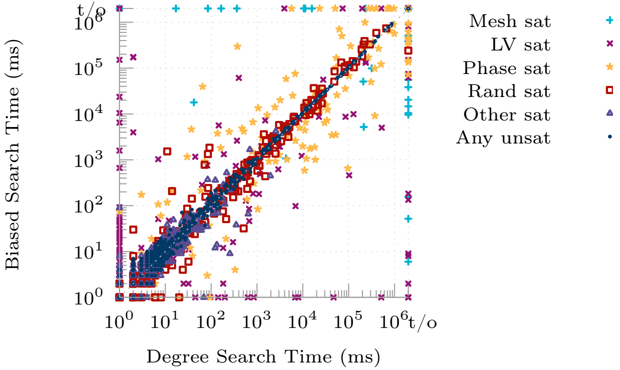
$$p(v') = \frac{2^{\deg(v')}}{\sum_{w \in D_v} 2^{\deg(w)}}.$$

- Looks a lot like *softmax*, which uses base e.

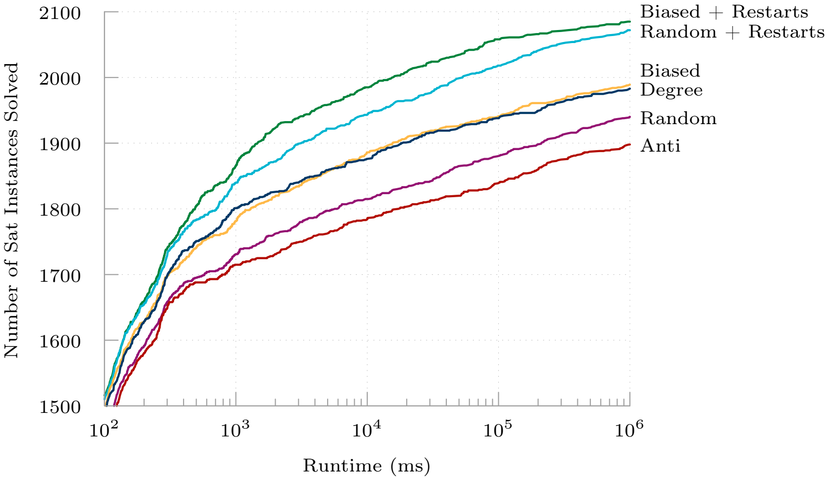
Biased Value-Ordering



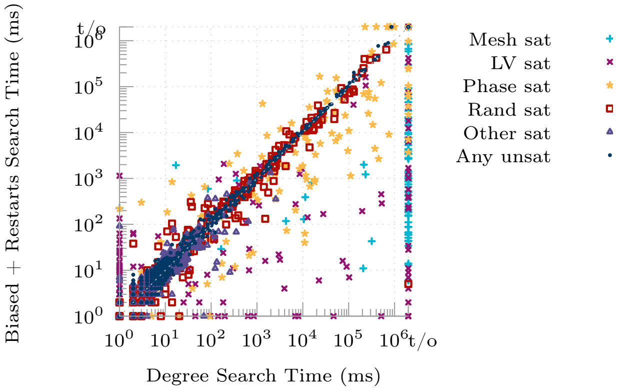
Biased Value-Ordering



Biased Value-Ordering with Restarts



Biased Value-Ordering with Restarts



Ongoing Work

- This works well for maximum common subgraph too.
- What about CP in general?
- Parallel!

http://www.dcs.gla.ac.uk/~ciaran
ciaran.mccreesh@glasgow.ac.uk

University of Glasgow