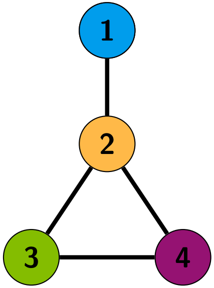
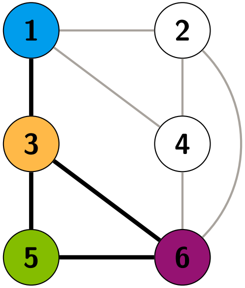
Solving Hard Subgraph Problems in Parallel

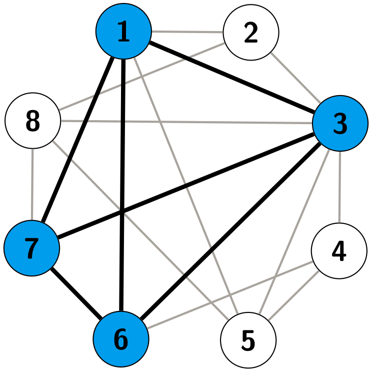
Ciaran McCreesh and Patrick Prosser

Subgraph Isomorphism

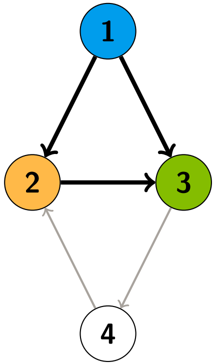


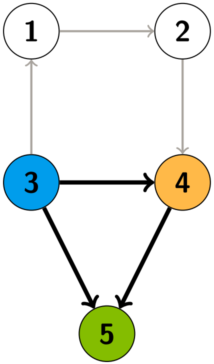


The Maximum Clique Problem

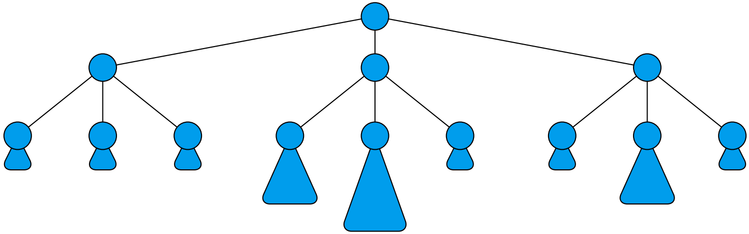


Maximum Common Subgraph

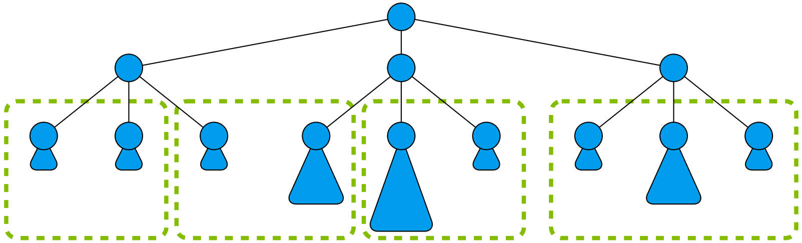




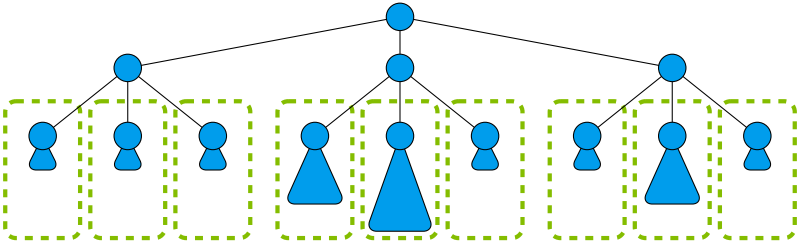
Thread-Parallel Tree Search



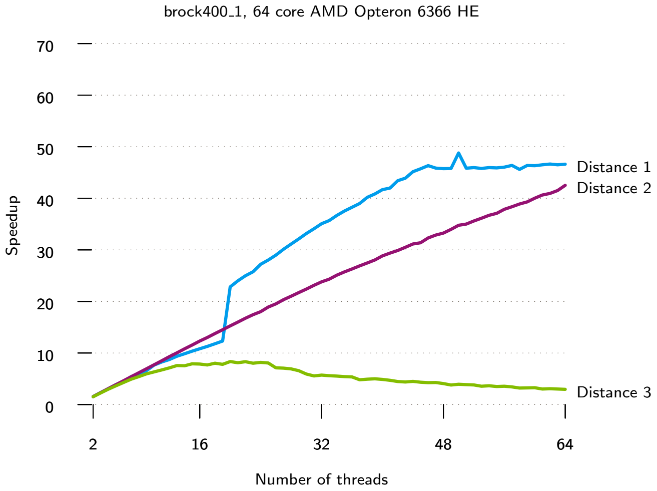
Thread-Parallel Tree Search



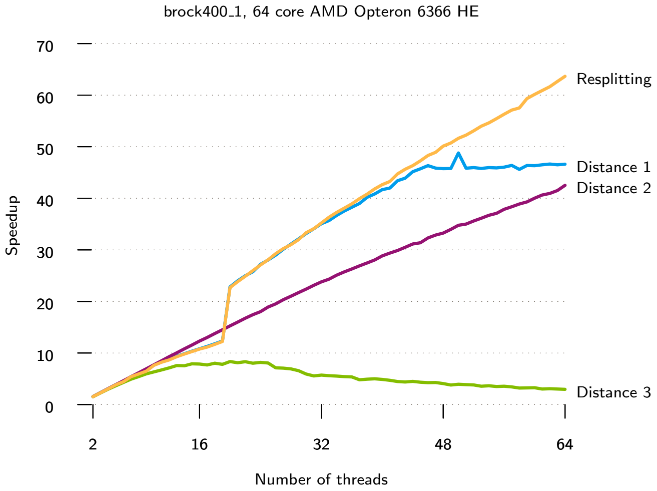
Thread-Parallel Tree Search



Parallel Search Order Matters



Parallel Search Order Matters



Safety and Reproducibility in Parallel Search

- My “wish list”:
  1. Parallel search should **not be substantially slower** than sequential search.
  2. Adding more processors should **not make things substantially worse**.
  3. Running the same program twice on the same hardware should give **similar runtimes**.
- This is surprisingly tricky.
- On top of all that, we want to **prioritise work stealing** from where we’re most likely to be wrong.

Graph Algorithms and Optimisation

- There are a lot of real-world optimisation problems involving **a graph problem** (subgraph isomorphism, subgraph covering, finding sequences of related subgraphs, clique finding, graph colouring, . . .), plus some **other constraints**.
- Can we make these problems easier to specify in a **high-level constraint modelling language** like MiniZinc?
- There is a continuum of what we could do with these models:
  - Compile to CP, MIP or SAT (but these models tend to be large, and lose structural and heuristic information).
  - A hybrid, multi-solver approach, "graph morphisms modulo theories" style (but we need better theories).
  - Compile to subgraph isomorphism (but even simple arithmetic constraints become disgusting under reduction).

http://www.dcs.gla.ac.uk/~ciaran

c.mccreesh.1@research.gla.ac.uk