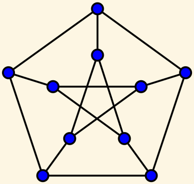


Symmetry Breaking in the Subgraph Isomorphism Problem

WHOOPS Workshop 2024

Joseph Loughney

jpl9@st-andrews.ac.uk

Graph Isomorphism

Graphs G and H are *isomorphic* if there exists a bijection between their vertex sets that preserves adjacency.

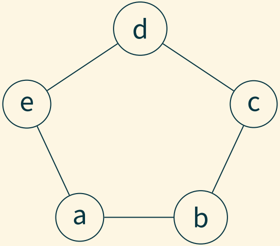


Figure: The graph G.

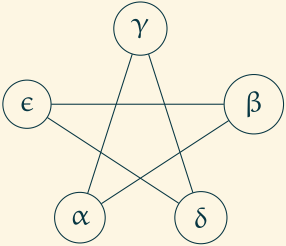


Figure: The (isomorphic) graph H.

Subgraph Isomorphism Problem (SIP)

Given a pattern graph P and a (larger) target graph T, does there exist a graph isomorphism between P and a subgraph of T?

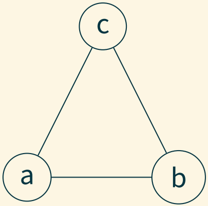


Figure: The pattern graph P.

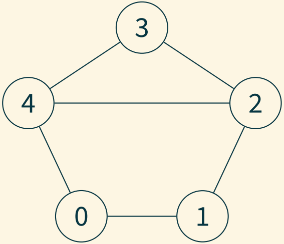


Figure: The target graph T.

Subgraph Isomorphism Problem (SIP)

Given a pattern graph P and a (larger) target graph T, does there exist a graph isomorphism between P and a subgraph of T?

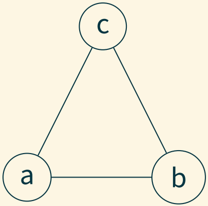


Figure: The pattern graph P.

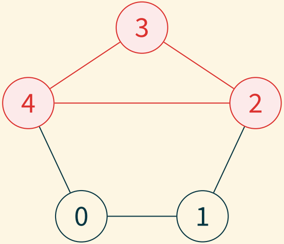


Figure: The target graph T.

Subgraph Isomorphism Problem (SIP)

How many such solutions exist?

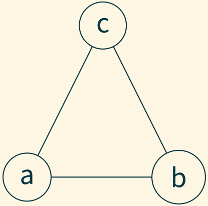


Figure: The pattern graph P.

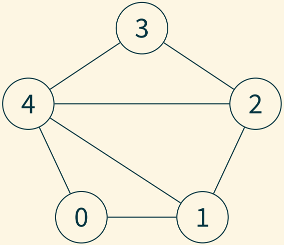


Figure: The target graph T.

Subgraph Isomorphism Problem (SIP)

How many such solutions exist?

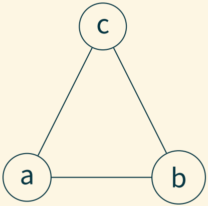
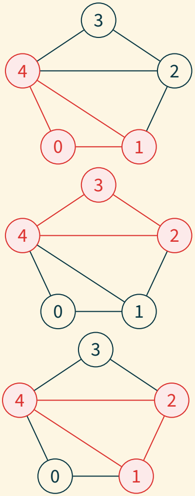
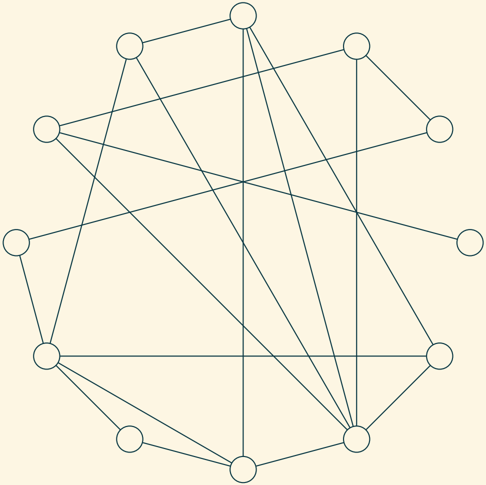


Figure: The pattern graph *P*.



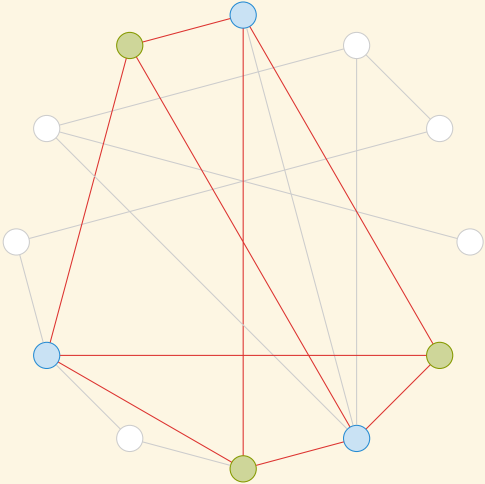
A Larger Example

Is this graph planar?

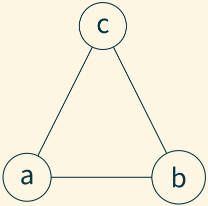


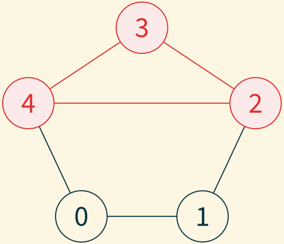
A Larger Example

No! We've found $K_{3,3}$ as a SIP solution, so G is non-planar (by Kuratowski's Theorem).



Symmetry Breaking





> ***Solutions***
> $(a \to 2, b \to 3, c \to 4)$, $(a \to 2, b \to 4, c \to 3)$,
> $(a \to 3, b \to 2, c \to 4)$, $(a \to 3, b \to 4, c \to 2)$,
> $(a \to 4, b \to 2, c \to 3)$, $(a \to 4, b \to 3, c \to 2)$

How do we avoid wasting time finding *symmetrical* solutions?

Pattern Symmetries

An isomorphism of a graph onto itself is called an *automorphism*.

Lemma

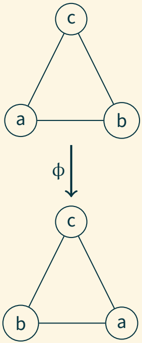
If an automorphism maps pattern vertex $p_1 \rightarrow p_2$, and a solution exists that maps p_1 to target vertex t, then an equivalent solution exists for $p_2 \rightarrow t$.

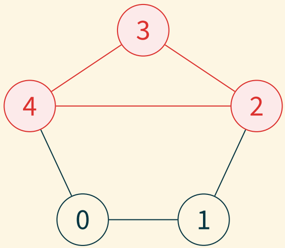
The set of all vertices $\{p_1, \cdots, p_j\}$ such that p maps to all p_i under automorphism is called the *orbit* of p, denoted $orb(p)$.

Pattern Symmetries

Example

$\phi(a) = b$, Solution 1 = $(a \to 2, b \to 3, c \to 4) \implies \exists$ Solution 2 = $(b \to 2, \cdots)$



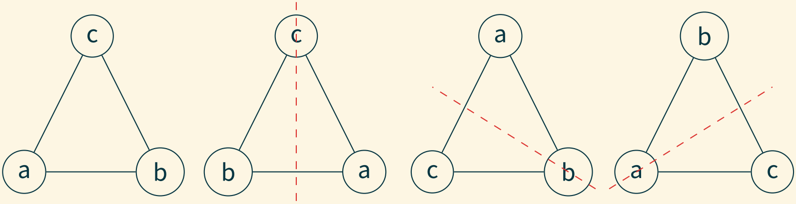


Solution 2

$(b \to 2, a \to 3, c \to 4)$

Pattern Symmetries

Consider the symmetries of the pattern graph, and the constraints we generate from them:



$a < b$ $\quad$ $a < c$ $\quad$ $b < c$

Pattern Constraints

Question

What does the pattern constraint $a < b$ mean?

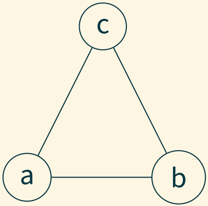
Answer

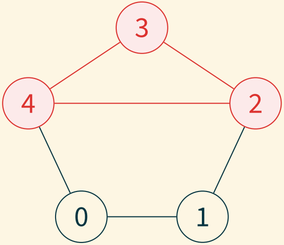
"The value assigned to variable a must be less than the value assigned to variable b."*

*More on this later...

Pattern Symmetries

Now using the variable constraints $a < b$, $a < c$, $b < c$:





> ***Solutions***
> $(a \rightarrow 2, b \rightarrow 3, c \rightarrow 4)$, $(a \rightarrow 2, b \rightarrow 4, c \rightarrow 3)$,
> $(a \rightarrow 3, b \rightarrow 2, c \rightarrow 4)$, $(a \rightarrow 3, b \rightarrow 4, c \rightarrow 2)$,
> $(a \rightarrow 4, b \rightarrow 2, c \rightarrow 3)$, $(a \rightarrow 4, b \rightarrow 3, c \rightarrow 2)$

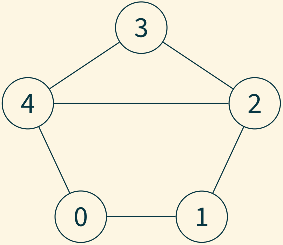
Target Symmetries

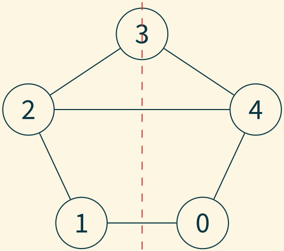
Similarly to pattern symmetries:

> ***Lemma***
> If there exists an automorphism which maps target vertex $t_1 \rightarrow t_2$, and a solution mapping $p \rightarrow t_1$, an equivalent solution exists mapping $p \rightarrow t_2$.

Target Symmetries

Consider the symmetries of the target:





$0 < 1$ or $2 < 4$

Pattern Constraints

Question

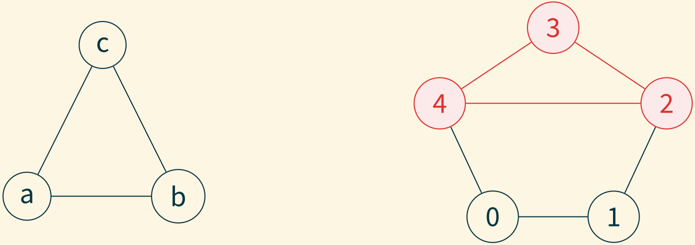
What does the target constraint $5 < 3$ mean?

Answer

"The value 5 must be assigned to a smaller variable than the value 3, if the value 3 is assigned to a variable."

Target Symmetries

Now using the value constraints $2 < 4$:



Solutions

$(a \to 2, b \to 3, c \to 4)(a \to 2, b \to 4, c \to 3)$,
$(a \to 3, b \to 2, c \to 4)$,$(a \to 3, b \to 4, c \to 2)$,
$(a \to 4, b \to 2, c \to 3)$, $(a \to 4, b \to 3, c \to 2)$

Target Symmetries

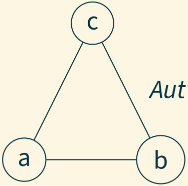
Note: Subgraph isomorphism is *non-surjective* on the target graph, so target constraints picked at random at the top of search may be ignored.

If we had picked $0 < 1$ instead of $2 < 4$, we wouldn't have filtered any solutions.

Generating Symmetry Constraints

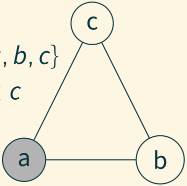
1. Compute generators for $Aut(G)$, the permutation group consisting of all automorphisms of the graph G.
2. For each generator g:
   a. Pick a base point β to stabilise, recording its orbit.
   b. Check which points still permute under g without moving β.
   c. Repeat steps *a* and *b* with one such point, if one exists.
3. For all non-trivial base points $\beta_1, \cdots, \beta_n$, add symmetry constraints $\beta_i < \gamma \; \forall \, \gamma \in orb(\beta_i)$.

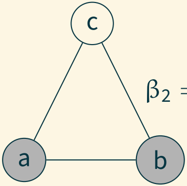
Example



$$Aut(G) = < (a\,b\,c), (a\,b) >$$

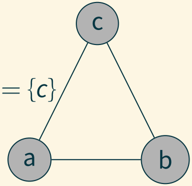
$$\beta_1 = a, orb(a) = \{a, b, c\}$$
$$\implies a < b, a < c$$





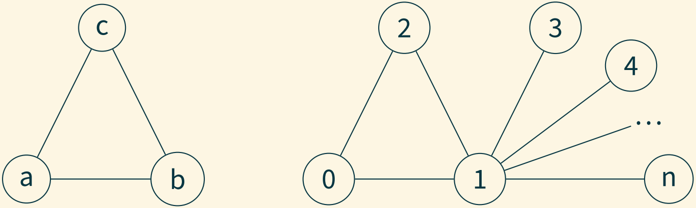
$$\beta_2 = b, orb(b)|_{\beta_1 = a} = \{b, c\}$$
$$\implies b < c$$

$$orb(c)|_{\beta_1 = a \wedge \beta_2 = b} = \{c\}$$



Flexible Ordering

Generating constraints before search may waste effort.



Using the fixed-order constraints generated at the top of search, we might propagate a list of constraints $\{3 < n, 4 < n, \cdots, n-1 < n\}$ after each assignment that is never relevant to search.

Flexible Ordering

> ***How do we combat this?***
> Fixed-order symmetry breaking picks a base at random before search. It could be unlucky, and pick one not relevant to the solution.

We can construct the base during search, adding constraints as we go. When we encounter a vertex not already in the base, we add it to the base. The "smallest" element in each orbit is simply the element we encounter first in the search tree.

Flexible Ordering

Suppose we have variable constraint $a < b$ and value constraint $4 < 2$, and find a solution with $(a \to 4, b \to 2, \cdots)$.

We want to accept this solution.

By constructing a *flexible value ordering* from the value constraints (such as $[1, 2, 3, 4, 5, \cdots] \to [1, 4, 2, 3, 5, \cdots]$ in this case), we can avoid accidentally adding conflicting variable-value constraint pairs.

Experimental Set-up

Experiments are run on the *Benchmarks for the Subgraph Isomorphism Problem* dataset found at

`https://perso.liris.cnrs.fr/christine.solnon/SIP.html`

Solver output is recorded, including: solution count, run-time, number of search nodes, and number of propagations.

Initial Results (Variable and Value Symmetry Breaking)

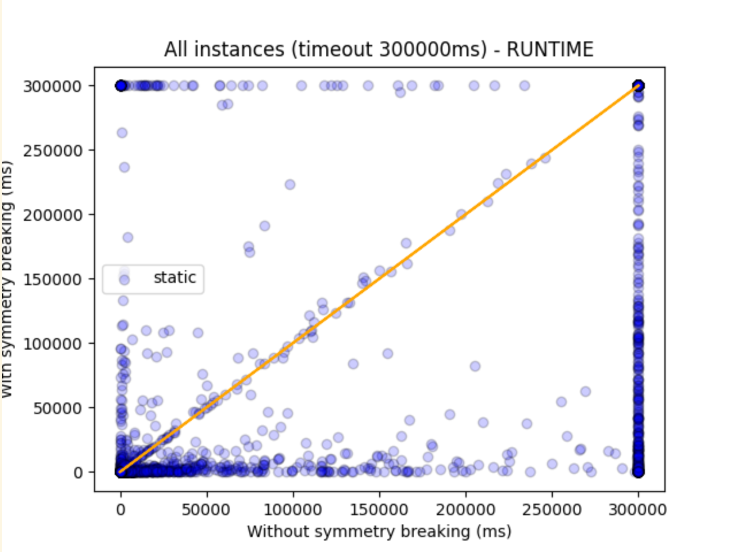


Figure: Fixed order symmetry breaking vs. No symmetry breaking

Initial Results (Variable and Value Symmetry Breaking)

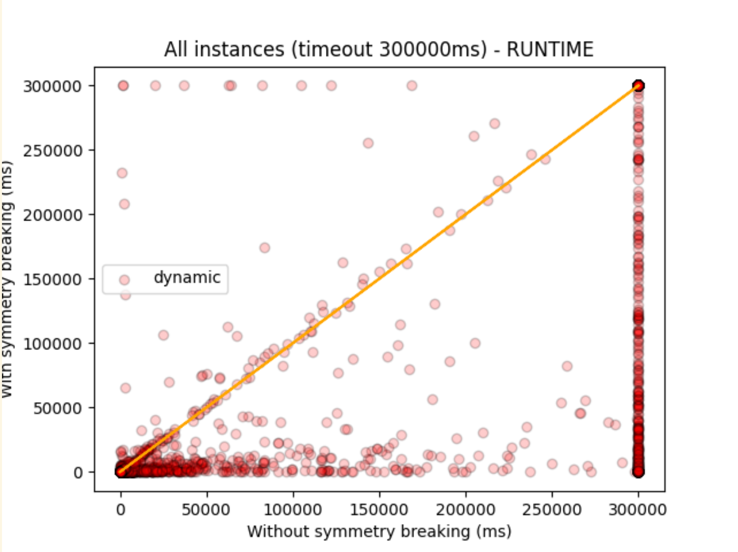


Figure: Flexible order symmetry breaking vs. No symmetry breaking

Initial Results (Variable and Value Symmetry Breaking)

Number of previously timed-out cases now solved, and vice versa

Of 17006 instances:	**Fixed Ordering**	**Flexible Ordering**
Now solved	178	213
Now timed-out	94	10

Average run-time relative to the solver with no symmetry breaking

Threshold (ms)	**Fixed Ordering**	**Flexible Ordering**
100	12.79	1.46
500	3.21	1.25
1000	2.00	1.10
5000	1.10	0.83
10000	1.07	0.84
50000	0.89	0.85

Initial Results (Variable and Value Symmetry Breaking)

Average relative run-time (unsatisfiable cases)

Threshold (ms)	Fixed Ordering	Flexible Ordering
100	22.37	1.96
500	18.10	4.25
1000	11.97	3.78
5000	3.93	1.03
10000	3.79	1.08
50000	1.35	0.87

Average relative run-time (satisfiable cases)

Threshold (ms)	Fixed Ordering	Flexible Ordering
100	0.53	0.57
500	0.47	0.33
1000	0.46	0.32
5000	0.46	0.26
10000	0.43	0.24
50000	0.34	0.20

Any questions?