Computing Maximum Unavoidable Subgraphs Using SAT Solvers

Cuong Chau & Marijn Heule

{ckcuong,marijn}@cs.utexas.edu

Department of Computer Science

The University of Texas at Austin

July 7, 2016

Outline

Outline

Introduction

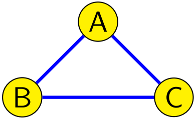
Definition 1 (Graph isomorphism)

Two graphs G and H are isomorphic if there exists an edge-preserving bijection from the vertices of G to the vertices of H. Isomorphic graphs occur in the same isomorphism class.

Definition 2 (Unavoidable subgraph)

A graph G is called an unavoidable subgraph (USG) of the **fully-connected graph** K_n if every red/blue edge-coloring of K_n contains an isomorphic graph of G **in only one color**.

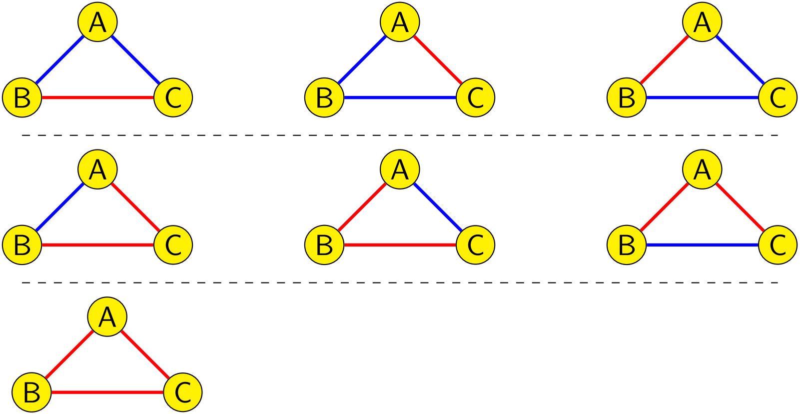
Introduction



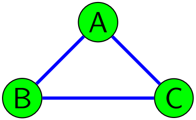
For all red/blue edge-colorings of K_3,

exists a monochromatic path of two edges.

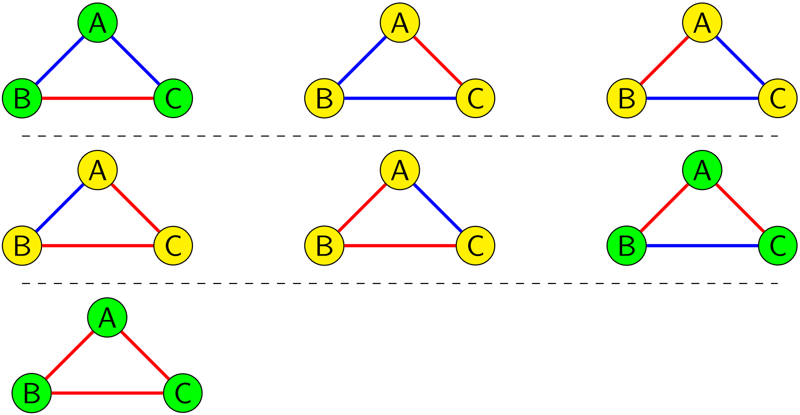
(Unavoidable subgraph)



Introduction



There exists a graph in each isomorphism class of K_3 s.t. the path B-A-C is monochromatic.



Motivation

Our approach tries to compute a maximum unavoidable subgraph (measured in the number of edges) for a given complete graph using SAT solvers.

Motivation

Our approach tries to compute a maximum unavoidable subgraph (measured in the number of edges) for a given complete graph using SAT solvers.

Unlike many **nicely structured** unavoidable subgraphs (USGs) (e.g., cliques, cycles, stars) that have been heavily studied, maximum USGs may not have a clear structure.

Motivation

Our approach tries to compute a maximum unavoidable subgraph (measured in the number of edges) for a given complete graph using SAT solvers.

Unlike many **nicely structured** unavoidable subgraphs (USGs) (e.g., cliques, cycles, stars) that have been heavily studied, maximum USGs may not have a clear structure.

USGs allow for an alternative symmetry-breaking approach for graph problems: given a USG, we can simplify graph problems by enforcing that all edges in the USG are either all present or all absent.

Motivation

Our approach tries to compute a maximum unavoidable subgraph (measured in the number of edges) for a given complete graph using SAT solvers.

Unlike many **nicely structured** unavoidable subgraphs (USGs) (e.g., cliques, cycles, stars) that have been heavily studied, maximum USGs may not have a clear structure.

USGs allow for an alternative symmetry-breaking approach for graph problems: given a USG, we can simplify graph problems by enforcing that all edges in the USG are either all present or all absent.

The larger the USG (measured in the number of edges), the stronger the symmetry-breaking predicate (SBP) can be derived (explained in Section 4).

Outline

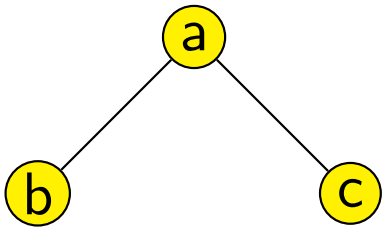
SAT Encoding of USGs

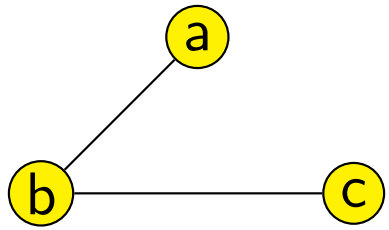
We employ a SAT solver to check whether a given graph G of order k is a USG of K_n ($k \leq n$).

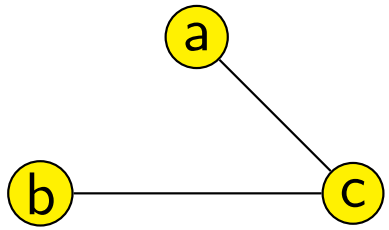
SAT Encoding of USGs

We employ a SAT solver to check whether a given graph G of order k is a USG of K_n ($k \leq n$).

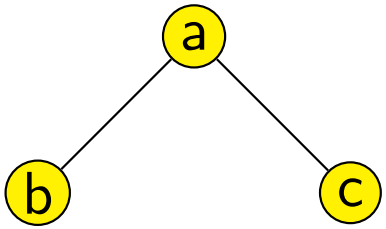
Encoding: Let's see how we encode the USG problem into SAT through the following example: *Check if a path of two edges is a USG of* K_3.

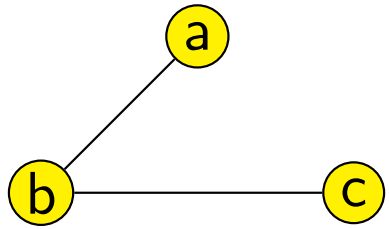


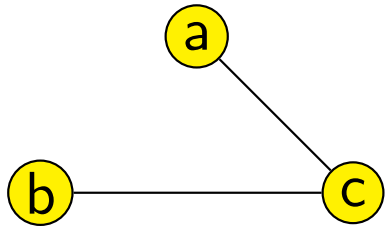




SAT Encoding of USGs







Let ***ab***, ***ac***, and ***bc*** denote the **Boolean variables representing the color of the edge** connecting vertices a and b, a and c, and b and c, respectively. If a Boolean variable has value T, the corresponding edge has color red. Otherwise it has color blue.

$$\mathcal{F} = (ab \wedge ac) \vee (\overline{ab} \wedge \overline{ac}) \vee (ab \wedge bc) \vee (\overline{ab} \wedge \overline{bc}) \vee (ac \wedge bc) \vee (\overline{ac} \wedge \overline{bc})$$

A path of two edges is a USG of K_3.
$\Leftrightarrow \mathcal{F}$ is VALID.
$\Leftrightarrow \overline{\mathcal{F}}$ is UNSATISFIABLE.

$$\overline{\mathcal{F}} = (\overline{ab} \vee \overline{ac}) \wedge (ab \vee ac) \wedge (\overline{ab} \vee \overline{bc}) \wedge (ab \vee bc) \wedge (\overline{ac} \vee \overline{bc}) \wedge (ac \vee bc)$$

Since $\overline{\mathcal{F}}$ is in CNF, SAT solvers can solve it directly.

Computing USGs Mechanically

SAT encoding of a USG problem (construct $\overline{\mathcal{F}_{G,K_n}}$ – referred to as $\overline{\mathcal{F}}$ in the previous slide):

Computing USGs Mechanically

SAT encoding of a USG problem (construct $\overline{\mathcal{F}_{G,K_n}}$ – referred to as $\overline{\mathcal{F}}$ in the previous slide):
For each subgraph H of K_n that is **isomorphic** to G, construct the following two clauses:

- **disjunction** of positive literals representing red color of edges in H,
- **disjunction** of negative literals representing blue color of edges in H.

Computing USGs Mechanically

SAT encoding of a USG problem (construct $\overline{\mathcal{F}_{G,K_n}}$ – referred to as $\overline{\mathcal{F}}$ in the previous slide):
For each subgraph H of K_n that is **isomorphic** to G, construct the following two clauses:

- **disjunction** of positive literals representing red color of edges in H,
- **disjunction** of negative literals representing blue color of edges in H.

$\overline{\mathcal{F}_{G,K_n}}$ is the **conjunction** of all of these clauses.

Computing USGs Mechanically

SAT encoding of a USG problem (construct $\overline{\mathcal{F}_{G,K_n}}$ – referred to as $\overline{\mathcal{F}}$ in the previous slide):
For each subgraph H of K_n that is **isomorphic** to G, construct the following two clauses:

- **disjunction** of positive literals representing red color of edges in H,
- **disjunction** of negative literals representing blue color of edges in H.

$\overline{\mathcal{F}_{G,K_n}}$ is the **conjunction** of all of these clauses.

Our method computes USGs mechanically using a SAT solver in combination with the tool nauty [B. McKay and A. Piperno, 2014] (for automatically generating input graphs) and symmetry-breaking methods.

Computing USGs Mechanically

SAT encoding of a USG problem (construct $\overline{\mathcal{F}_{G,K_n}}$ – referred to as $\overline{\mathcal{F}}$ in the previous slide):
For each subgraph H of K_n that is **isomorphic** to G, construct the following two clauses:

- **disjunction** of positive literals representing red color of edges in H,
- **disjunction** of negative literals representing blue color of edges in H.

$\overline{\mathcal{F}_{G,K_n}}$ is the **conjunction** of all of these clauses.

Our method computes USGs mechanically using a SAT solver in combination with the tool nauty [B. McKay and A. Piperno, 2014] (for automatically generating input graphs) and symmetry-breaking methods.

A graph G is unavoidable in $K_n \Leftrightarrow$ UNSAT$(\overline{\mathcal{F}_{G,K_n}} \wedge SBP(\overline{\mathcal{F}_{G,K_n}}))$.

Reducing the Maximum USG Search Space

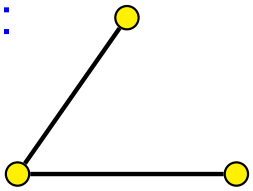
We detect the lower bound on the size and upper bound on the maximum degree of maximum USGs.

$\Rightarrow$ Using these bounds to reduce the maximum USG search space.

n	3	4	5	6	7	8	9
# isomorphism classes	4	11	34	156	1,044	12,346	274,668
# checked graphs	2	2	6	35	97	291	904

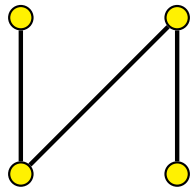
USG Results

K_3, K_4:



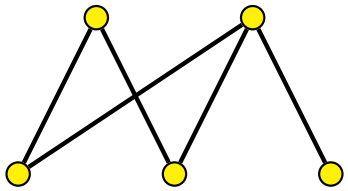
$|E| = 2$

K_5:



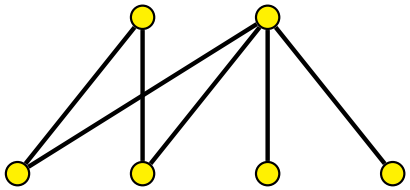
$|E| = 3$

K_6:



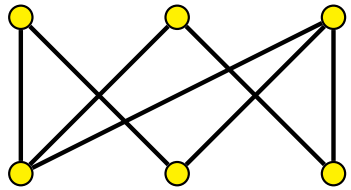
$|E| = 5$

K_7:



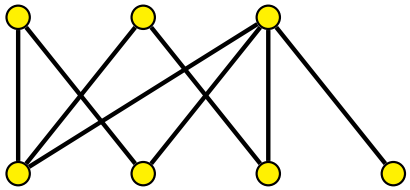
$|E| = 6$

K_8:



$|E| = 7$

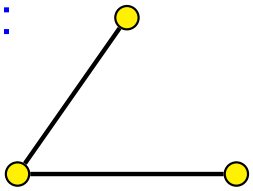
K_9:



$|E| = 8$

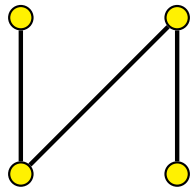
USG Results

K_3, K_4:



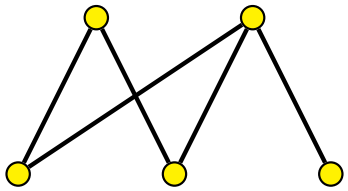
$|E| = 2$

K_5:



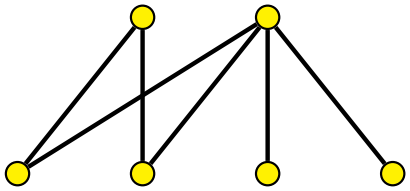
$|E| = 3$

K_6:



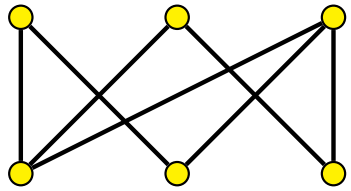
$|E| = 5$

K_7:



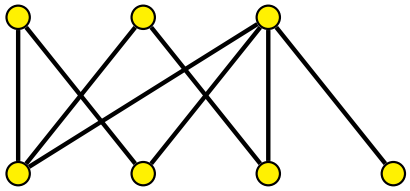
$|E| = 6$

K_8:



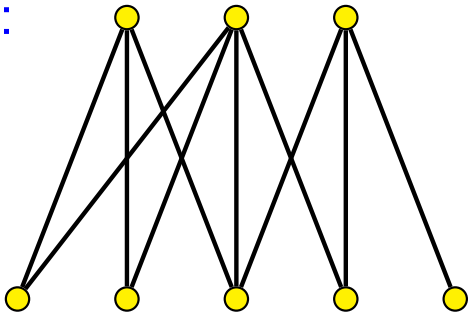
$|E| = 7$

K_9:



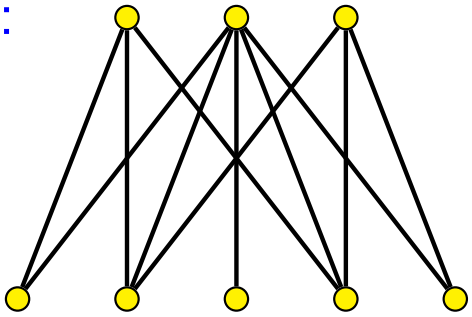
$|E| = 8$

K_{10}:



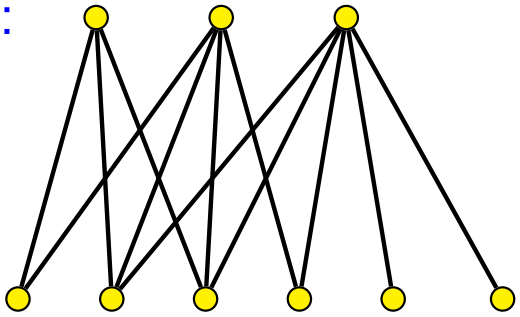
$|E| = 10$

K_{11}:



$|E| = 11$

K_{12}:



$|E| = 12$

Outline

Multi-Component USG

The concept USG can be generalized to multiple components, such that each component must occur monochromatic in all red/blue edge-colorings of K_n.

Multi-Component USG

The concept USG can be generalized to multiple components, such that each component must occur monochromatic in all red/blue edge-colorings of K_n.

We require that each component must have **at least two edges**.

Multi-Component USG

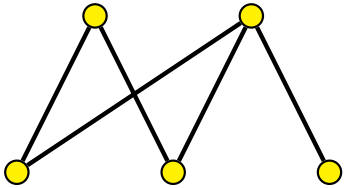
The concept USG can be generalized to multiple components, such that each component must occur monochromatic in all red/blue edge-colorings of K_n.

We require that each component must have **at least two edges**.

As heuristic to reduce the vast number of possible multi-component graphs, we restricted the search to graphs that have at most **three** components.

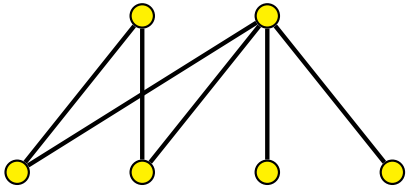
Multi-Component USG Results

K_6:



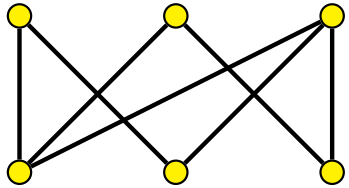
$|E| = 5$

K_7:

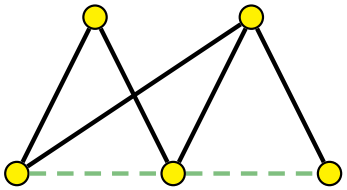


$|E| = 6$

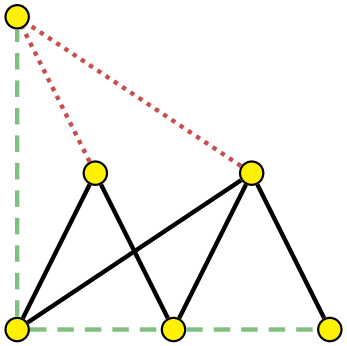
K_8:



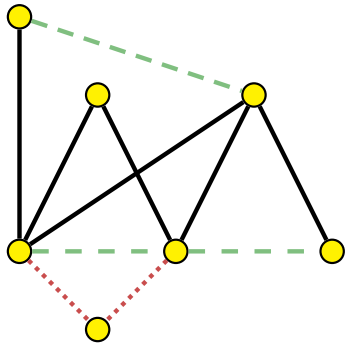
$|E| = 7$



$|E| = 7$



$|E| = 10$



$|E| = 11$

Outline

Deriving SBPs from USGs

Converting a USG G of K_n into an SBP by **forcing all edges in each component of G to be either all present or all absent**.

Deriving SBPs from USGs

Converting a USG G of K_n into an SBP by **forcing all edges in each component of G to be either all present or all absent**.

Let e_1, e_2,..., e_ℓ denote the Boolean variables representing the edges of one component in G.

$$e_1 \leftrightarrow e_2 \leftrightarrow e_3 \leftrightarrow \cdots \leftrightarrow e_\ell \quad \equiv \quad e_1 \rightarrow e_2 \rightarrow e_3 \rightarrow \cdots \rightarrow e_\ell$$

$$\equiv \quad (\overline{e_1} \vee e_2) \wedge (\overline{e_2} \vee e_3) \wedge \ldots \wedge (\overline{e_\ell} \vee e_1)$$

Deriving SBPs from USGs

Converting a USG G of K_n into an SBP by **forcing all edges in each component of G to be either all present or all absent**.

Let e_1, e_2,..., e_ℓ denote the Boolean variables representing the edges of one component in G.

$$e_1 \leftrightarrow e_2 \leftrightarrow e_3 \leftrightarrow \cdots \leftrightarrow e_\ell \quad \equiv \quad e_1 \rightarrow e_2 \rightarrow e_3 \rightarrow \cdots \rightarrow e_\ell$$

$$\equiv \quad (\overline{e_1} \vee e_2) \wedge (\overline{e_2} \vee e_3) \wedge \ldots \wedge (\overline{e_\ell} \vee e_1)$$

Each monochromatic component of ℓ edges can be encoded as a CNF formula consisting of ℓ binary clauses.

Deriving SBPs from USGs

Converting a USG G of K_n into an SBP by **forcing all edges in each component of G to be either all present or all absent**.

Let e_1, e_2,..., e_ℓ denote the Boolean variables representing the edges of one component in G.

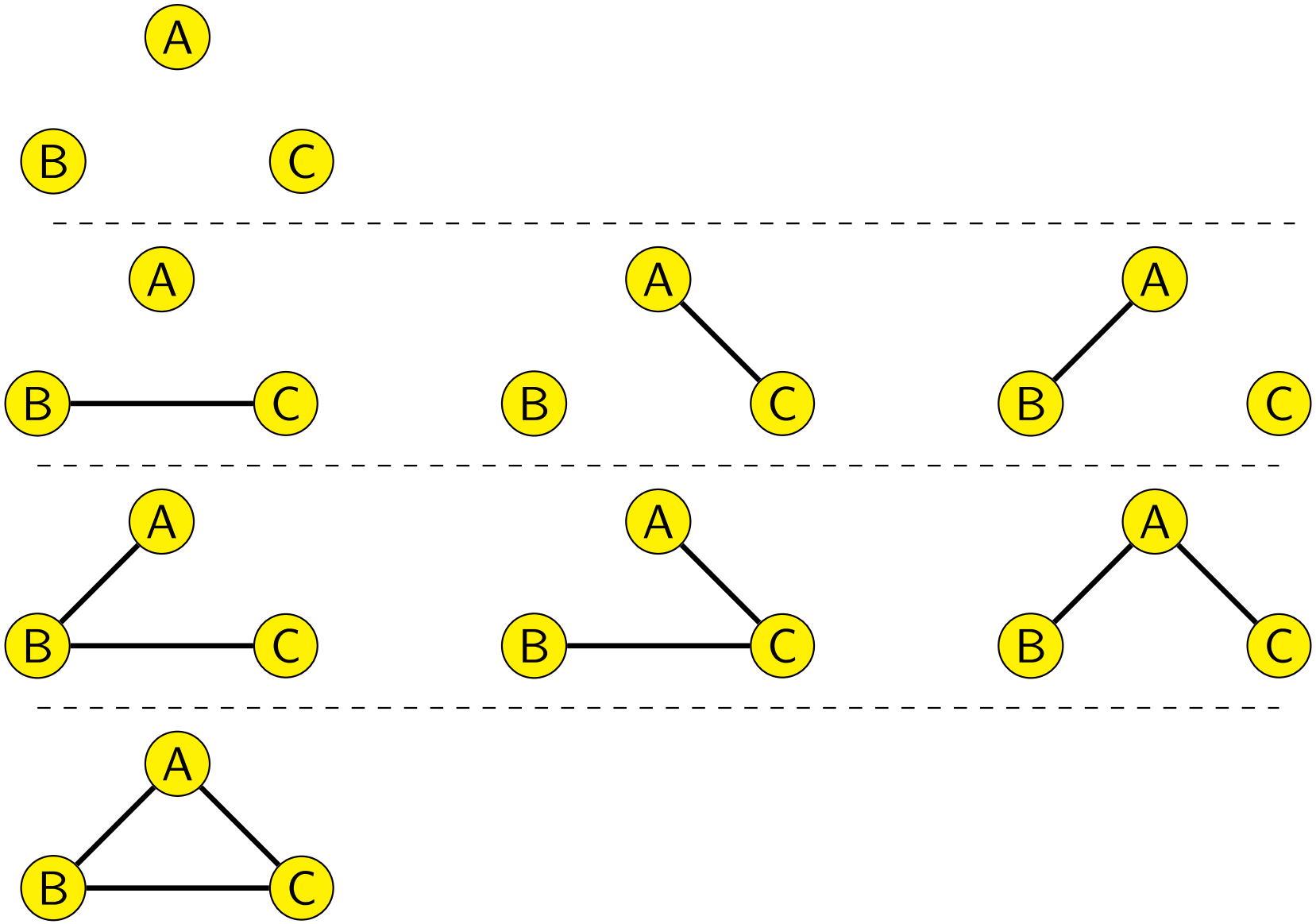
$$e_1 \leftrightarrow e_2 \leftrightarrow e_3 \leftrightarrow \cdots \leftrightarrow e_\ell \quad \equiv \quad e_1 \rightarrow e_2 \rightarrow e_3 \rightarrow \cdots \rightarrow e_\ell$$

$$\equiv \quad (\overline{e_1} \vee e_2) \wedge (\overline{e_2} \vee e_3) \wedge \ldots \wedge (\overline{e_\ell} \vee e_1)$$

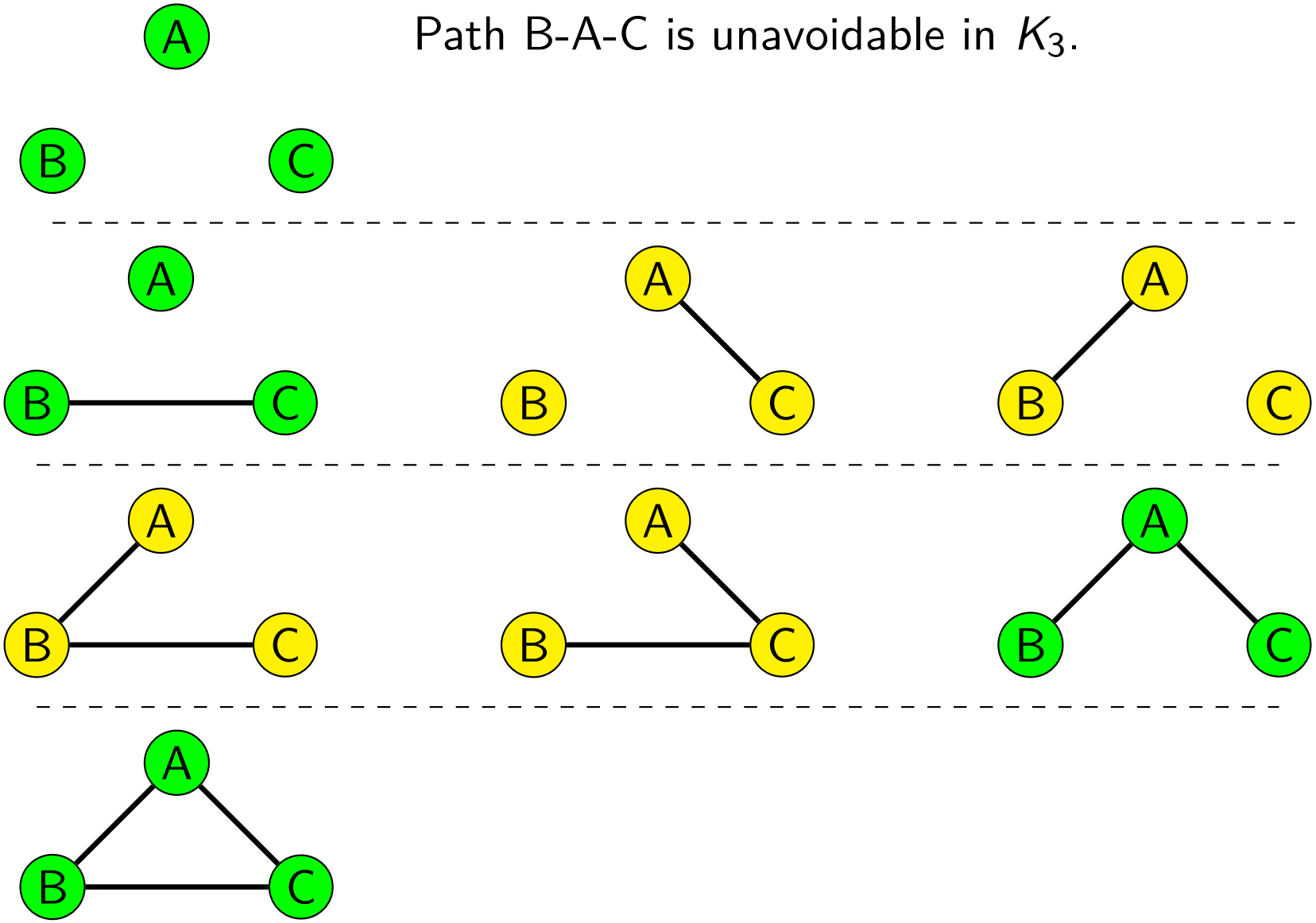
Each monochromatic component of ℓ edges can be encoded as a CNF formula consisting of ℓ binary clauses.

Applying this encoding for all components of G results in an SBP of $|E(G)|$ binary clauses.

Example



Example



How Useful Are the USG-Based SBPs?

The number of graphs of order n is $2^{|E_{K_n}|}$, where

$$|E_{K_n}| = \binom{n}{2} = \frac{n(n-1)}{2}$$

How Useful Are the USG-Based SBPs?

The number of graphs of order n is $2^{|E_{K_n}|}$, where

$$|E_{K_n}| = \binom{n}{2} = \frac{n(n-1)}{2}$$

Applying the SBP derived from a USG G of m components, the search space is reduced to $2^{|E_{K_n}|-|E_G|+m}$.

How Useful Are the USG-Based SBPs?

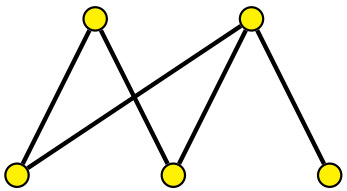
The number of graphs of order n is $2^{|E_{K_n}|}$, where

$$|E_{K_n}| = \binom{n}{2} = \frac{n(n-1)}{2}$$

Applying the SBP derived from a USG G of m components, the search space is reduced to $2^{|E_{K_n}|-|E_G|+m}$.
$\Rightarrow$ The larger the USG (measured in the number of edges), the stronger the SBP can be derived.

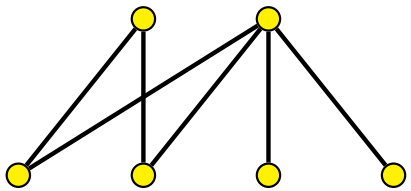
Single vs. Multi-Component USGs

K_6: $2^{|E(K_6)|} = 2^{15}$



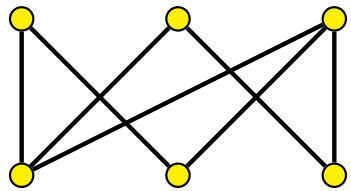
$2^{|E(K_6)|-5+1} = 2^{11}$

K_7: $2^{|E(K_7)|} = 2^{21}$

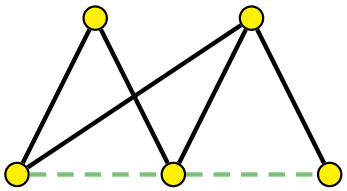


$2^{|E(K_7)|-6+1} = 2^{16}$

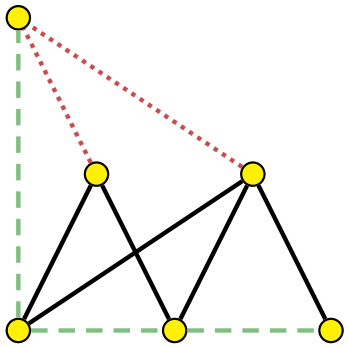
K_8: $2^{|E(K_8)|} = 2^{28}$



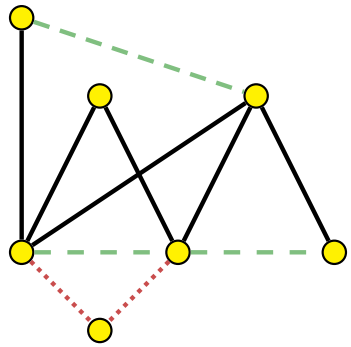
$2^{|E(K_8)|-7+1} = 2^{22}$



$2^{|E(K_6)|-7+2} = 2^{10}$



$2^{|E(K_7)|-10+3} = 2^{14}$



$2^{|E(K_8)|-11+3} = 2^{20}$

Runtime Results

Table: Runtime (in seconds) to compute the maximum or largest found single-component USGs of K_6 to K_{12} using the `glucose 3.0` SAT solver [G. Audemard and L. Simon, 2009]. The experiments were run on 3.5GHz Intel Xeon E31280 processors with 8MB L3 cache size. A '-' means a timeout after 24 hours.

n	6	7	8	9	10	11	12
No SBP	0	0.025	0.38	4.85	11,690.70	-	-
USG-SBP	0	0.002	**0.01**	**0.07**	26.45	266.94	532.48
L-SBP	0	**0**	**0.01**	0.11	**4.77**	**18.73**	312.60
Q-SBP	0	**0**	**0.01**	0.11	7.98	19.40	**303.40**

USG-SBP: SBP derived from the largest found single-component USG of K_{n-1}.

L-SBP: Linear SBP [F. Aloul et al., 2003].

Q-SBP: Quadratic SBP [M. Codish et al., 2013].

Outline

Conclusions

We have presented a method for computing maximum USGs mechanically via SAT solving and demonstrated how USGs can be converted into SBPs.

Conclusions

We have presented a method for computing maximum USGs mechanically via SAT solving and demonstrated how USGs can be converted into SBPs.

We observe that all maximum USGs for small graphs (up to K_9) are bipartite and conjecture that this holds in general.

Conclusions

We have presented a method for computing maximum USGs mechanically via SAT solving and demonstrated how USGs can be converted into SBPs.

We observe that all maximum USGs for small graphs (up to K_9) are bipartite and conjecture that this holds in general.

It appears that the maximum USGs of K_{n+1} are strictly larger than the maximum USGs of K_n for $n > 3$.

Conclusions

We have presented a method for computing maximum USGs mechanically via SAT solving and demonstrated how USGs can be converted into SBPs.

We observe that all maximum USGs for small graphs (up to K_9) are bipartite and conjecture that this holds in general.

It appears that the maximum USGs of K_{n+1} are strictly larger than the maximum USGs of K_n for $n > 3$.

Symmetry breaking was crucial to obtain our results. However, current symmetry-breaking techniques are not strong enough to compute some relatively simple USG problems using SAT.

Conclusions

We have presented a method for computing maximum USGs mechanically via SAT solving and demonstrated how USGs can be converted into SBPs.

We observe that all maximum USGs for small graphs (up to K_9) are bipartite and conjecture that this holds in general.

It appears that the maximum USGs of K_{n+1} are strictly larger than the maximum USGs of K_n for $n > 3$.

Symmetry breaking was crucial to obtain our results. However, current symmetry-breaking techniques are not strong enough to compute some relatively simple USG problems using SAT.

We envision that knowledge about the maximum USGs, both the single and multi-component variants, could be a basis for novel symmetry-breaking techniques for SAT solvers.

References

Fadi A. Aloul and Karem A. Sakallah and Igor L. Markov (2003)

Efficient Symmetry Breaking for Boolean Satisfiability

The 18th International Joint Conference on Artificial Intelligence, 271–276.

Michael Codish and Alice Miller and Patrick Prosser and Peter J. Stuckey (2013)

Breaking Symmetries in Graph Representation

The 23rd International Joint Conference on Artificial Intelligence, 510–516.

Brendan D. McKay and Adolfo Piperno (2014)

Practical Graph Isomorphism, II

Journal of Symbolic Computation, 60, 94–112.

Gilles Audemard and Laurent Simon (2009)

Predicting Learnt Clauses Quality in Modern SAT Solvers

The 21st International Joint Conference on Artificial Intelligence, 399–404.

Questions?