A log-space algorithm for reachability in planar acyclic digraphs with few sources[1]

Derrick Stolee Chris Bourke N. V. Vinodchandran
University of Nebraska-Lincoln
{dstolee,cbourke,vinod}@cse.unl.edu

June 10, 2010

[1]This work was supported by the NSF grants CCF-0430991 and CCF-0830730.

The Reachability Problem

Definition

Given a graph G and vertices u, v, the reachability problem asks if v is reachable from u.

The Reachability Problem

Definition

Given a graph G and vertices u, v, the reachability problem asks if v is reachable from u.

The complexity of reachability in directed planar graphs is not completely understood.

L vs NL and Reachability

NL – Non-deterministic Log-space

L – Deterministic Log-space

L vs NL and Reachability

NL – Non-deterministic Log-space

Directed Graphs **(even in DAGs)**

L – Deterministic Log-space

L vs NL and Reachability

NL – Non-deterministic Log-space

Directed Graphs **(even in DAGs)**

L – Deterministic Log-space

Undirected Reach (Reingold 08)

L vs NL and Reachability

NL – Non-deterministic Log-space

Directed Graphs **(even in DAGs)**

UL – Unambiguous Log-space

Directed Planar Reach (B, Tewari, V 2009)

L – Deterministic Log-space

Undirected Reach (Reingold 08)

L vs NL and Reachability

NL – Non-deterministic Log-space

Directed Graphs **(even in DAGs)**

UL – Unambiguous Log-space

Directed Planar Reach (B, Tewari, V 2009)

L – Deterministic Log-space

Undirected Reach (Reingold 08)
Series-Parallel Graphs
(Jakoby, Liśkiewicv, Reischuk 2006; Jakoby, Tantau 2007)

L vs NL and Reachability

NL – Non-deterministic Log-space

Directed Graphs **(even in DAGs)**

UL – Unambiguous Log-space

Directed Planar Reach (B, Tewari, V 2009)

L – Deterministic Log-space

Undirected Reach (Reingold 08)
Series-Parallel Graphs
(Jakoby, Liśkiewicv, Reischuk 2006; Jakoby, Tantau 2007)
Single-Source Multiple-Sink Planar DAGs (SMPD)
(Allender, Barrington, Chakraborty, Datta, Roy 2009)

Results

Theorem (Main Theorem)

The reachability problem for planar directed acyclic graphs with $m = m(n)$ sources is decidable in deterministic $O(m + \log n)$ space

Corollary

The reachability problem for planar directed acyclic graphs with $O(\log n)$ sources is in L.

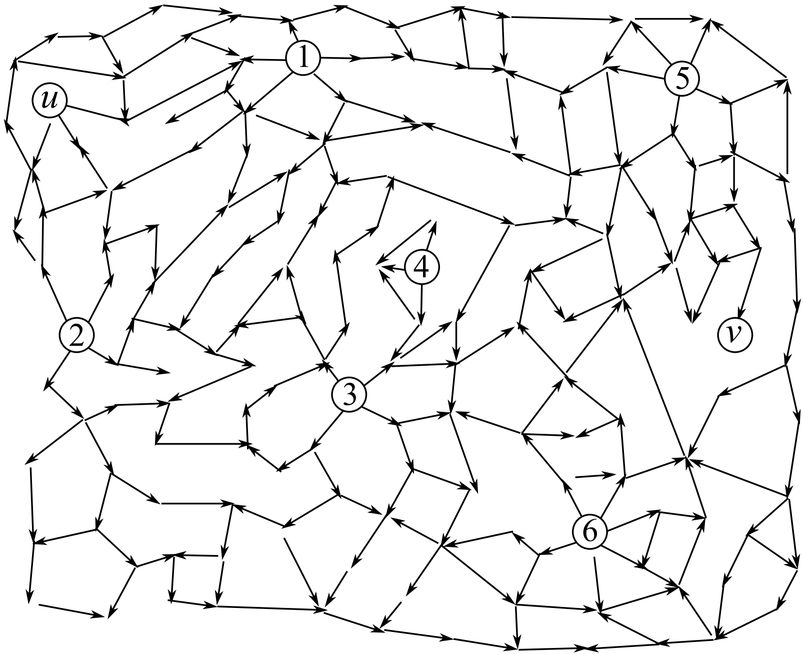
Proof Outline

Proof Outline

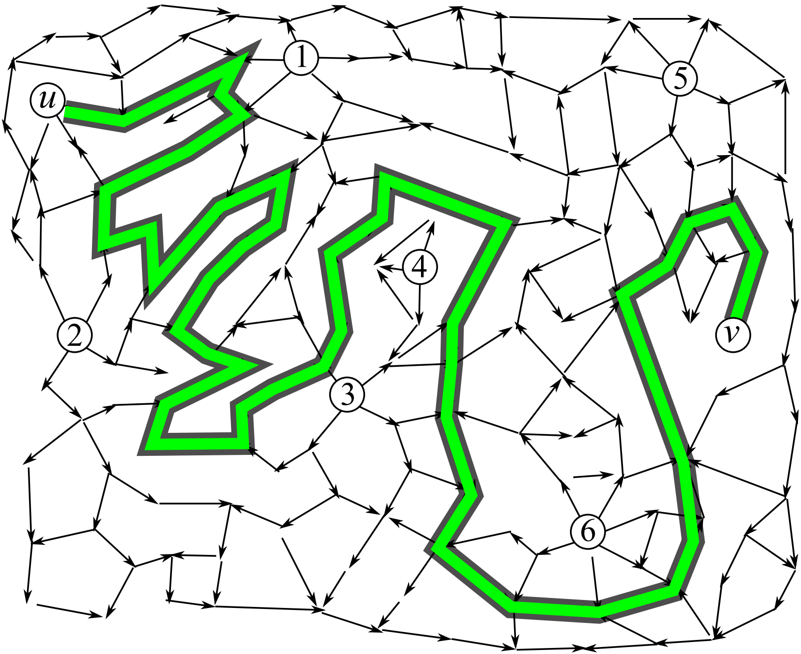
Let G be a planar DAG with vertices u, v, and m sources $s_1, \ldots, s_m$.

Definition (Forest Decomposition)

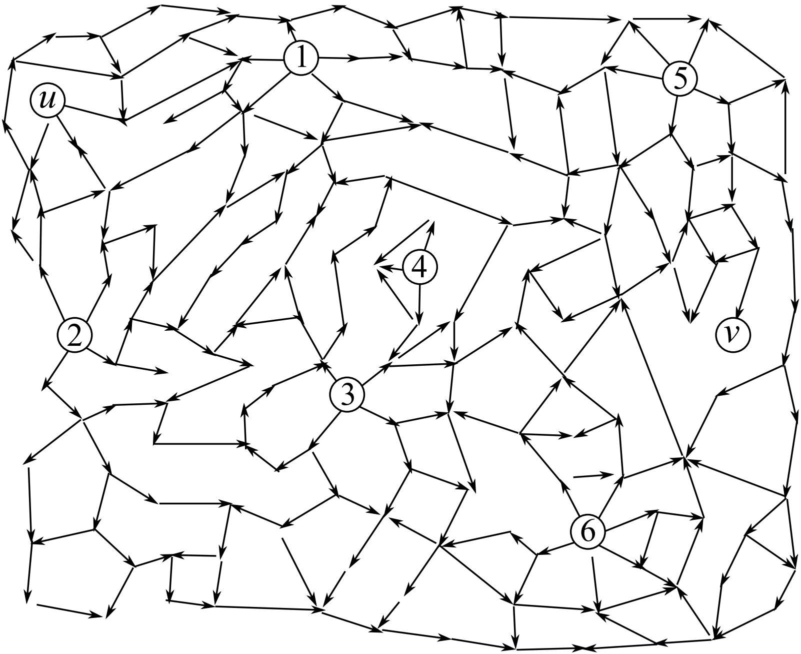
Select an incoming edge at each non-source vertex except u and v. The subgraph given by these edges is a **forest decomposition** F in G.

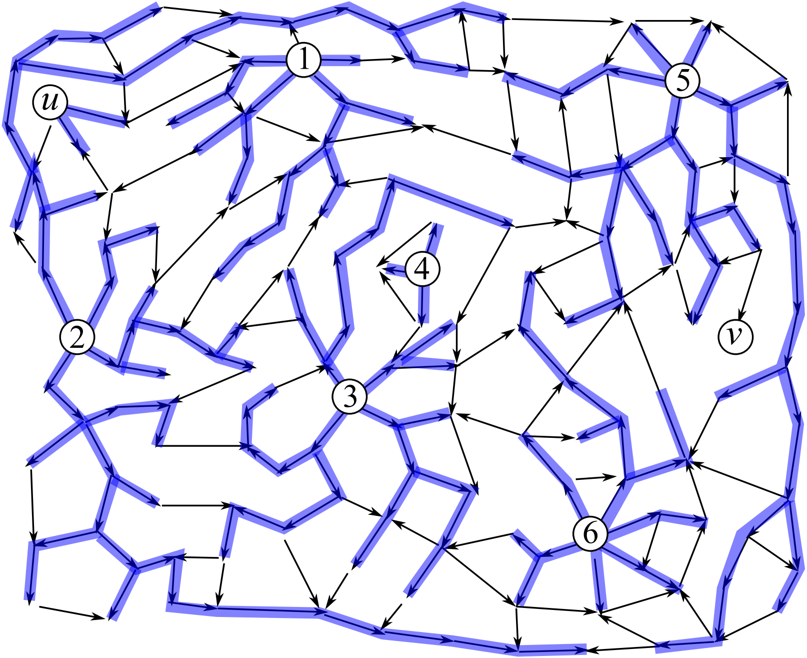


1
u
5
4
2
v
3
6



u
v
1
2
3
4
5
6


1
u
5
4
2
v
3
6



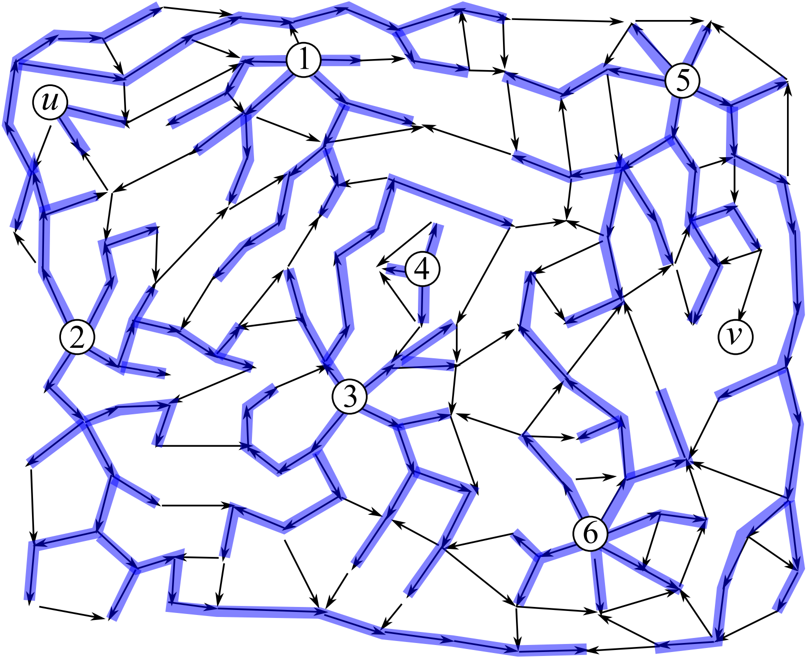
1
2
3
4
5
6
u
v

Contracted Graph: H

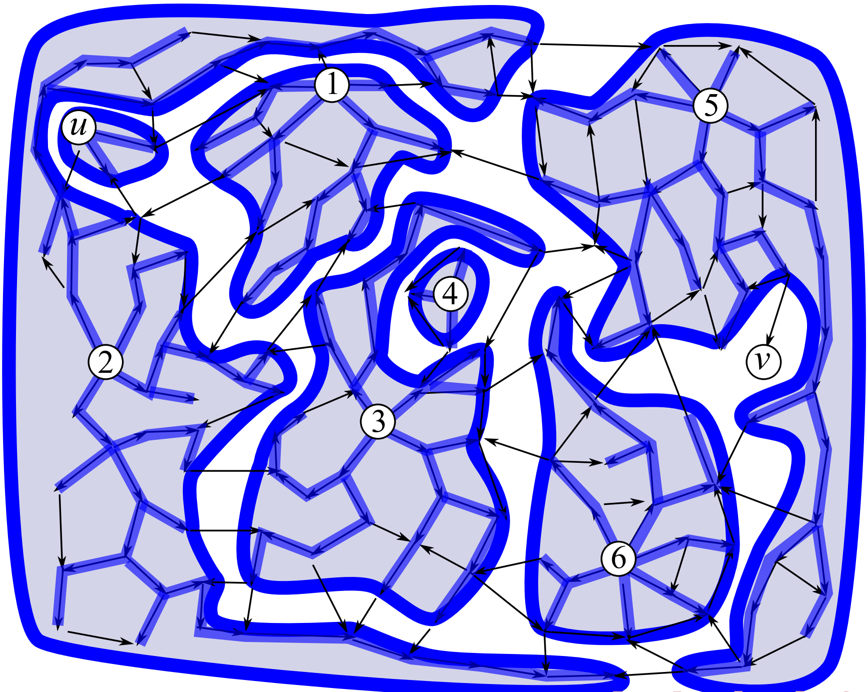
Definition (Contracted Graph)

Let H be the directed multigraph with $m + 2$ vertices given by contracting each tree in the forest F to the root vertex.

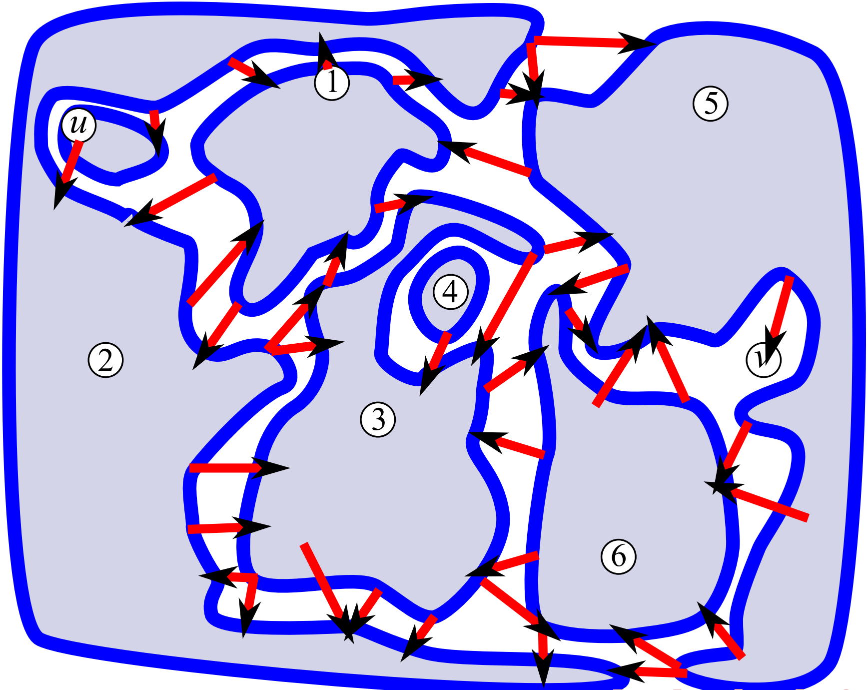
Call H the **contracted graph** of the decomposition F in G.



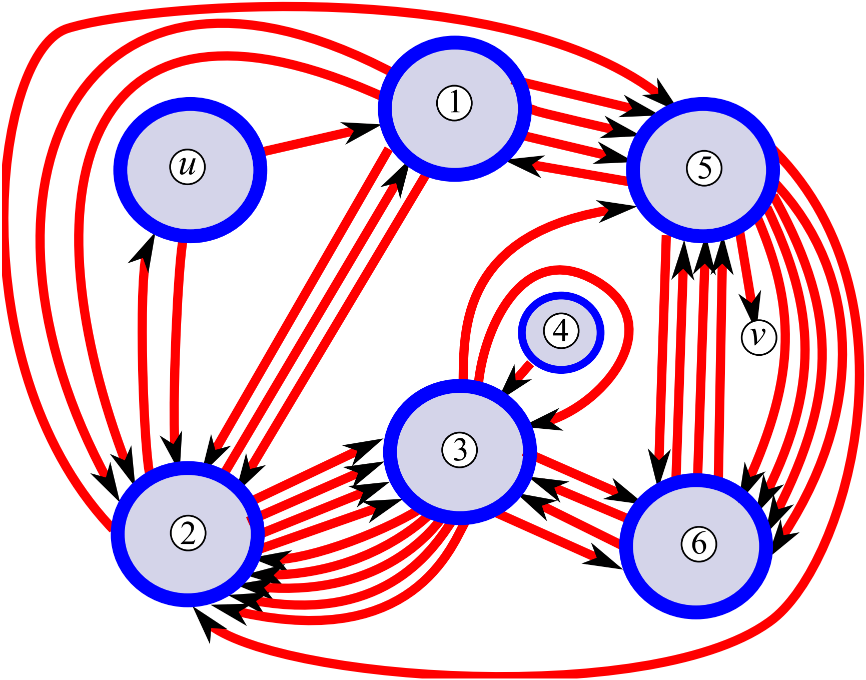
1
u
5
4
2
v
3
6



1
u
2
3
4
5
6
v

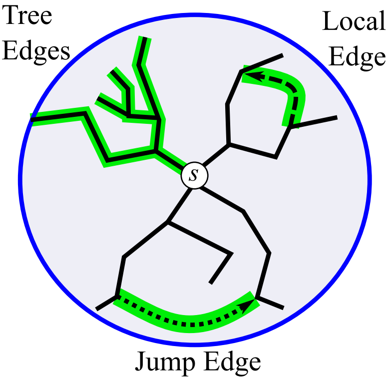


1
u
5
2
4
3
v
6



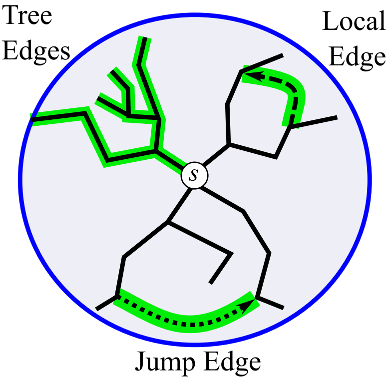
1
u
5
4
v
3
2
6

The SMPD Algorithm [ABCDR09]



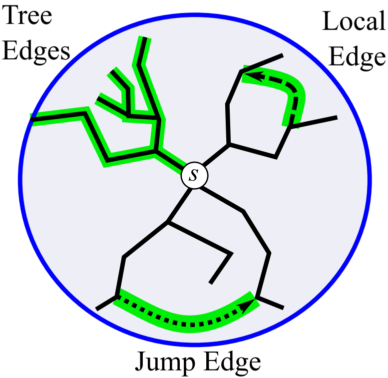
1. **Tree** edges are the edges in T.

The SMPD Algorithm [ABCDR09]

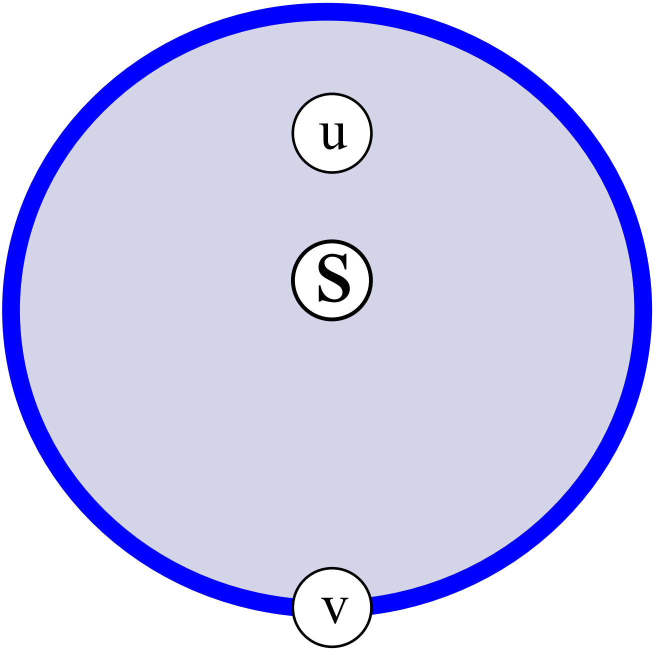


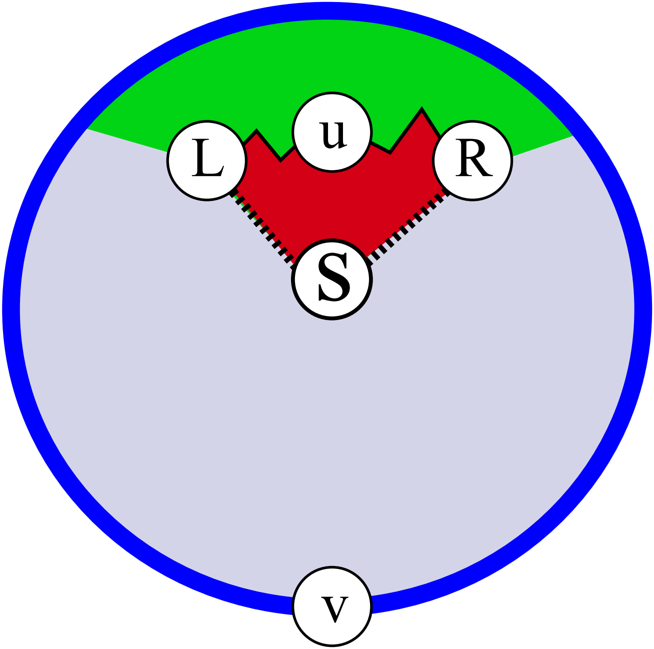
1. **Tree** edges are the edges in T.
2. **Local** edges enclose no leaves of T.

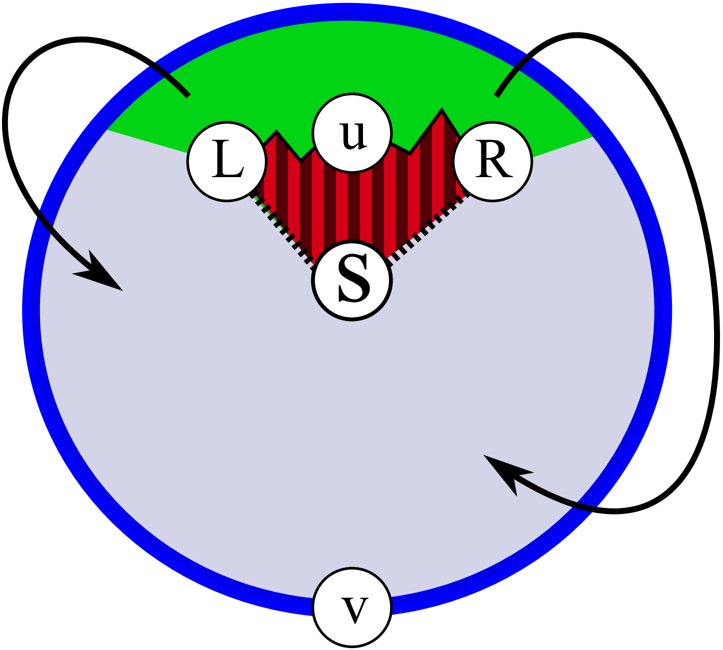
The SMPD Algorithm [ABCDR09]

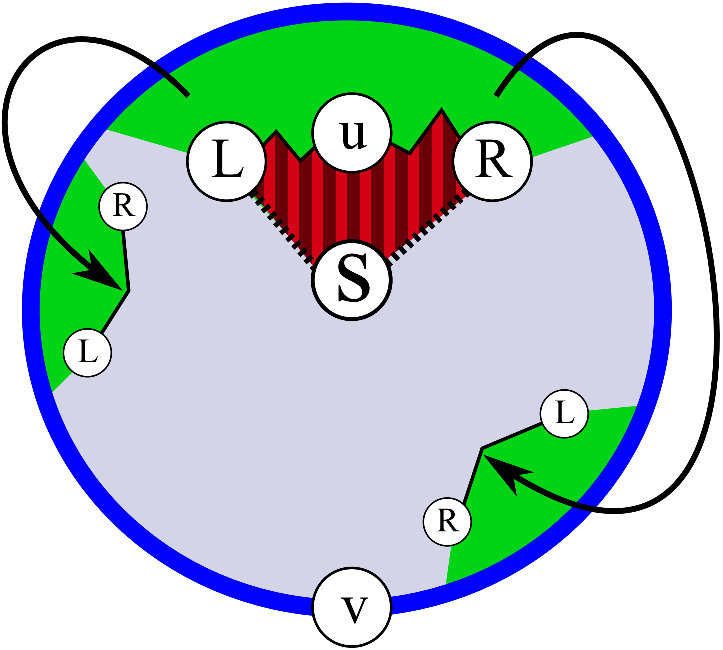


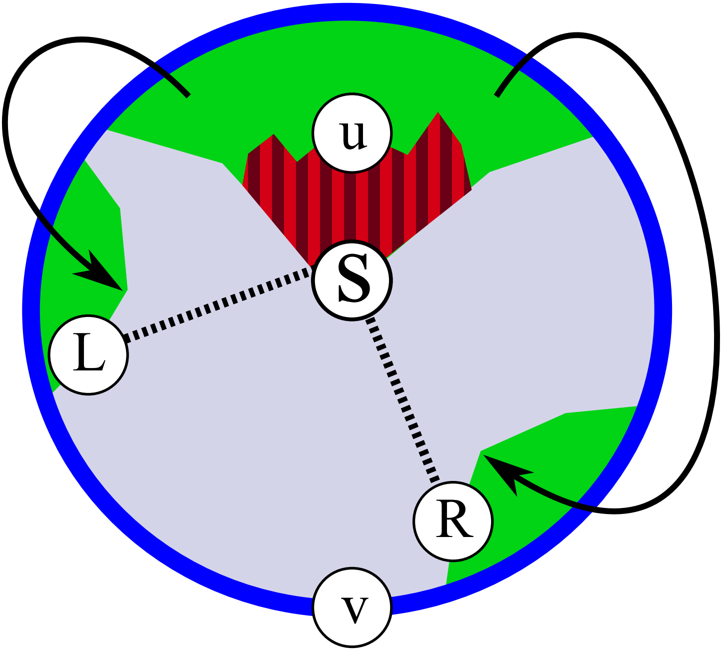
1. **Tree** edges are the edges in T.
2. **Local** edges enclose no leaves of T.
3. **Jump** edges enclose some leaves of T.

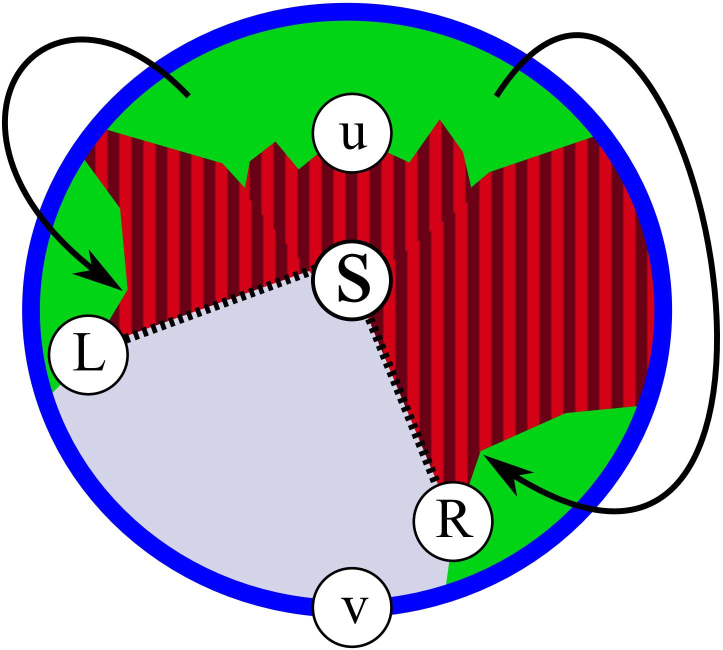

u
S
v

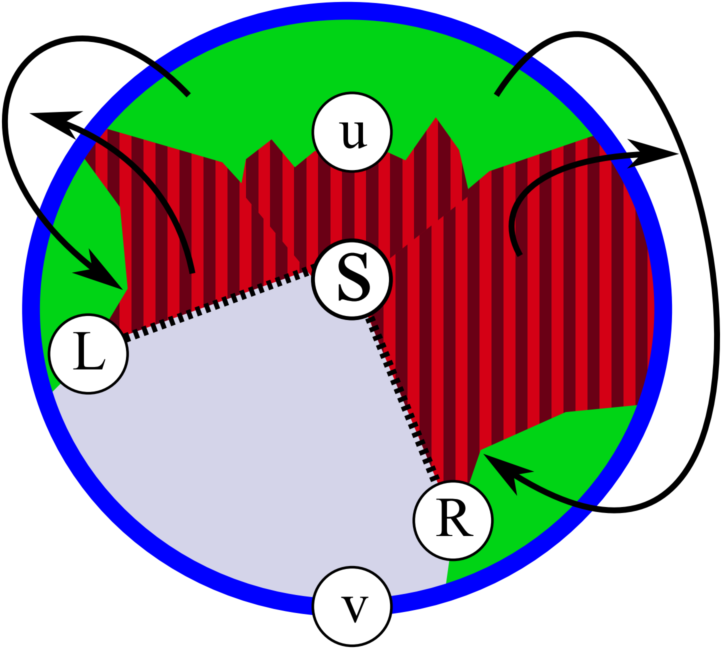

u
L
R
S
v


u
L
R
S
v

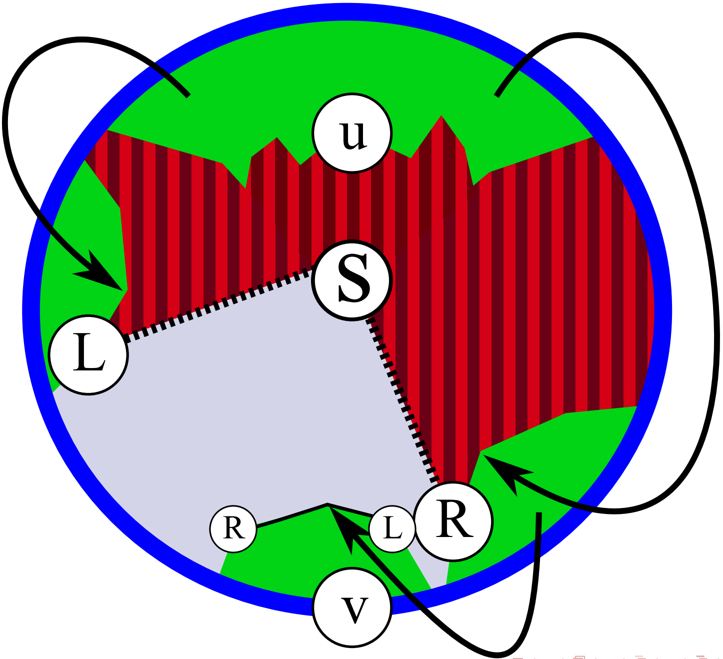

L
u
R
S
R
L
L
R
V

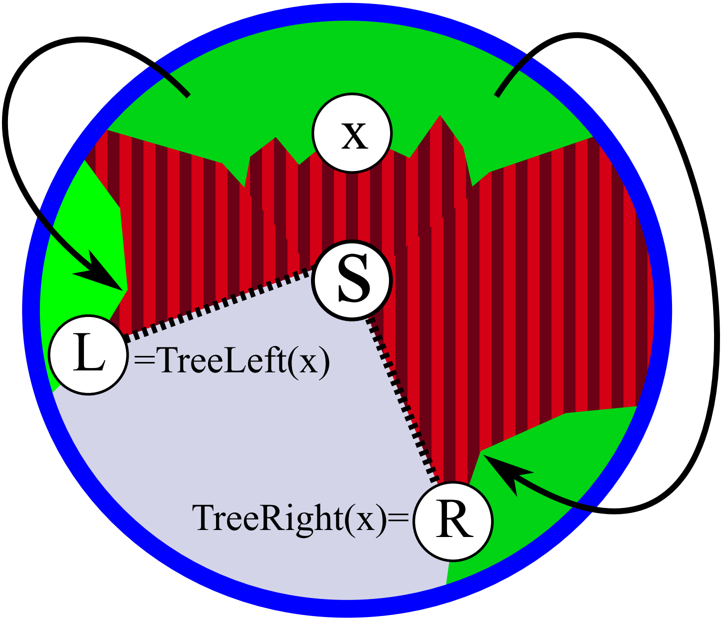

u
S
L
R
v


u
S
L
R
v



u
S
L
R
v

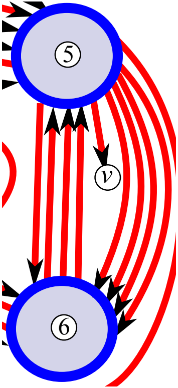

u
S
L
R
R
L
v


x
S
L
=TreeLeft(x)
TreeRight(x)=
R

New Edge Types

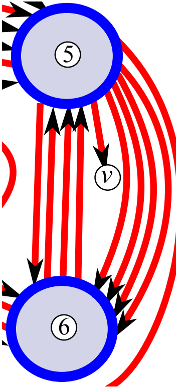
Launch Edges

span different source trees.

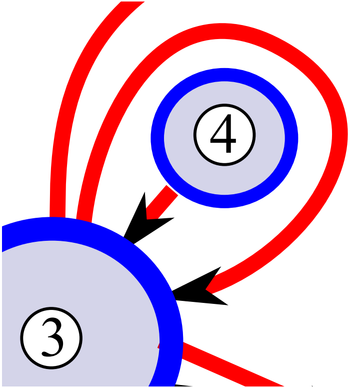


New Edge Types

Launch Edges
span different source trees.



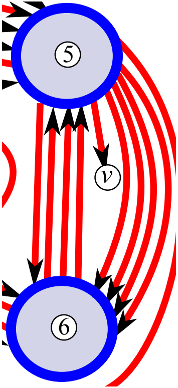
Loop edges
enclose entire source trees.



New Edge Types

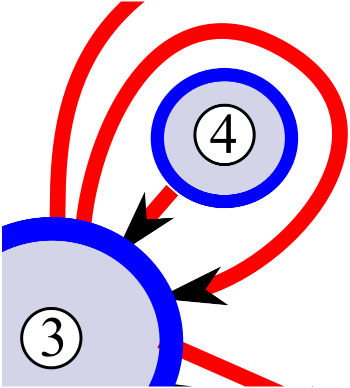
Launch Edges

span different source trees.



Loop edges

enclose entire source trees.



We require further classification of these edges!

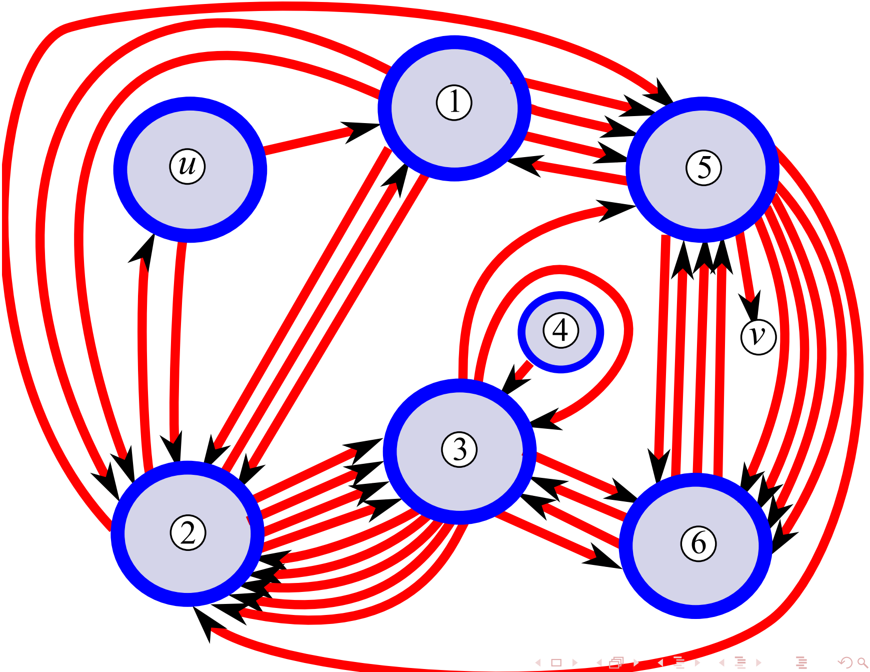
Proof Outline

Topological Equivalence

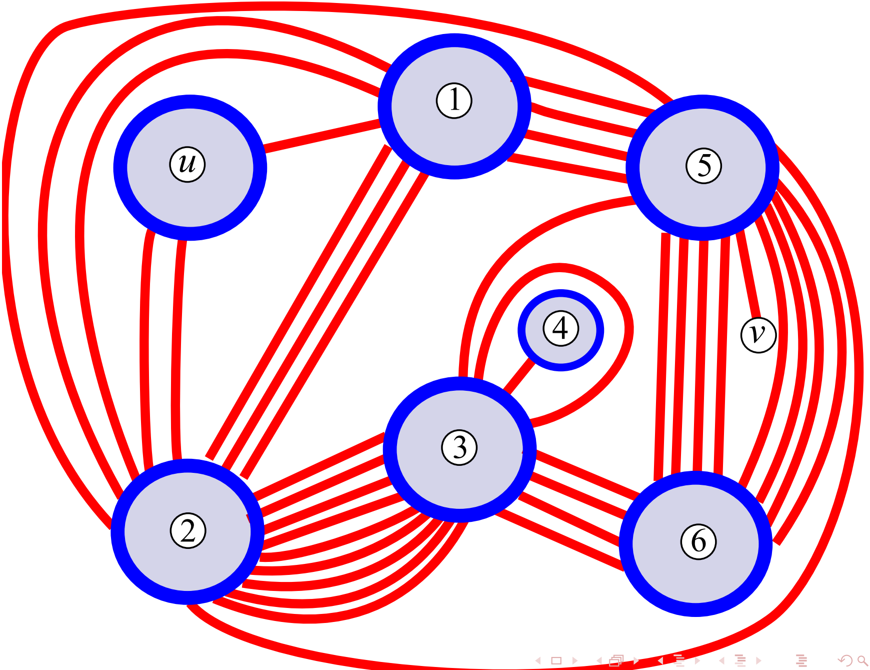
Let H be a multigraph embedded in the plane.

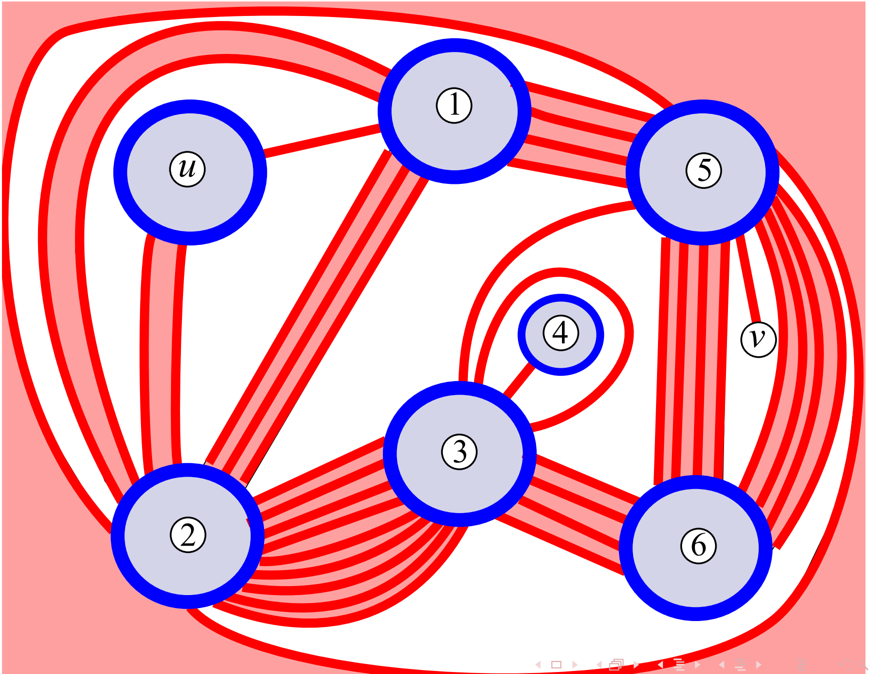
Definition

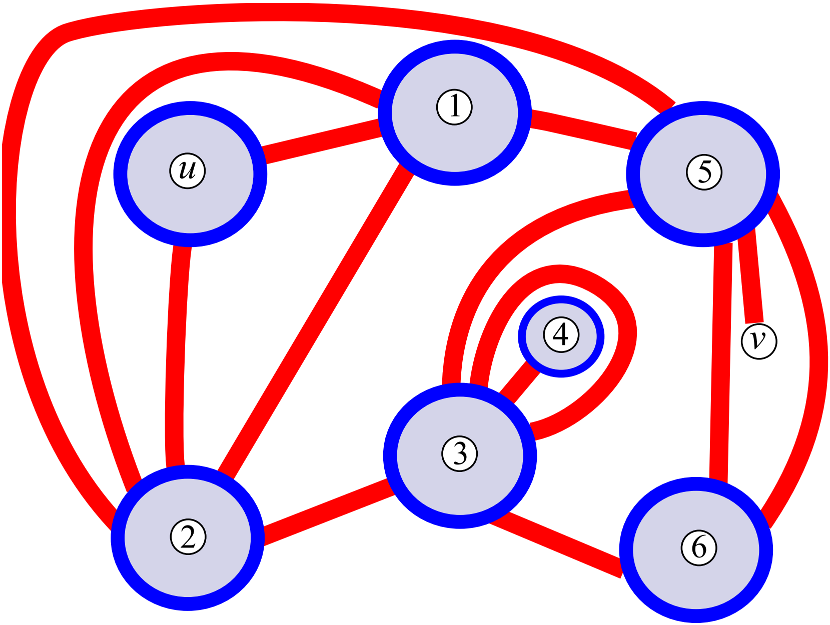
Two edges with common endpoints are **topologically equivalent** if the closed curve they form (in the underlying undirected graph) trivially partitions the other vertices.

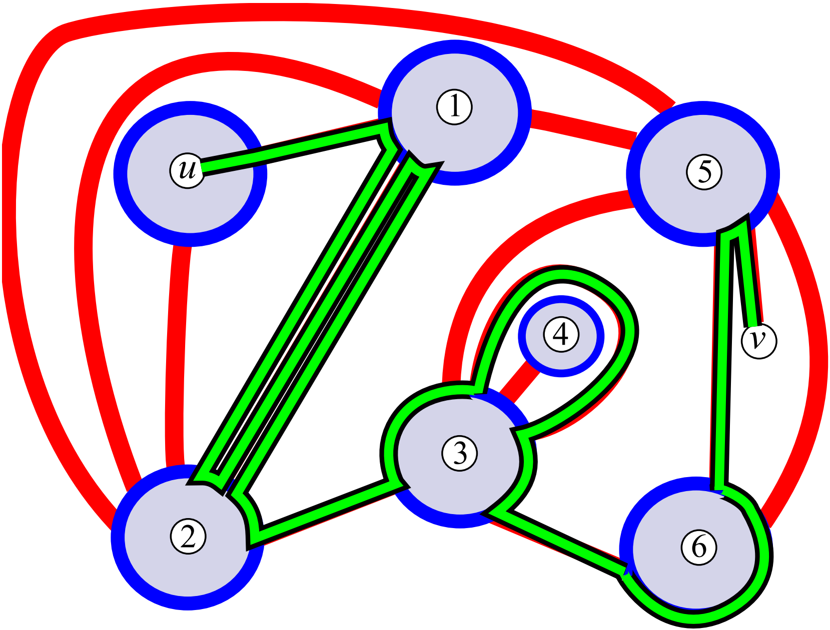


u
1
5
4
v
3
2
6


1
u
5
4
v
3
2
6


1
u
5
4
v
3
2
6


1
u
5
4
v
3
2
6



1
u
5
4
v
3
2
6

Euler's Formula and Class Bounds

Euler's formula holds for vertices, faces, and *equivalence classes*.

Lemma

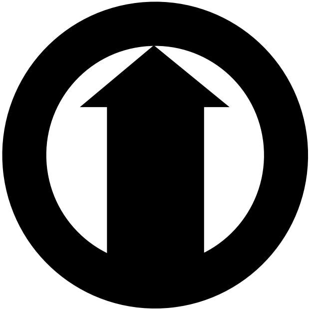
Let X be a planar multigraph with n_X vertices. Then X has at most $3n_X - 6$ equivalence classes of edges.

Hence, at most $3m$ classes in H (with $m + 2$ vertices).

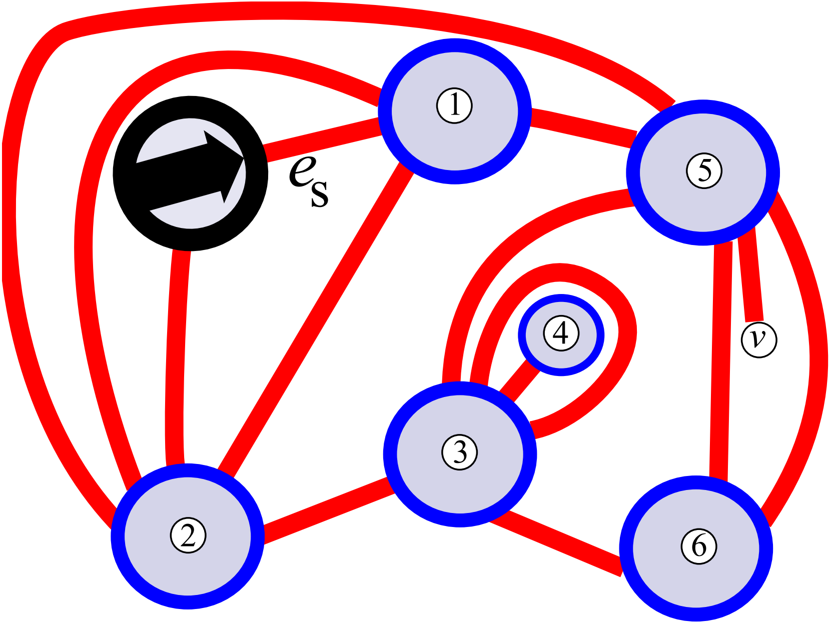
Proof Outline

The Coin-Crawl Game

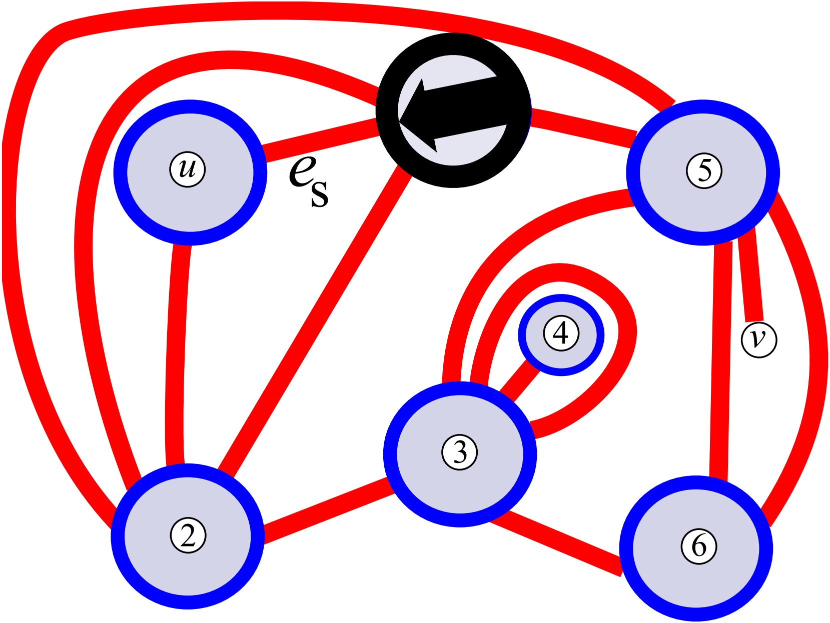
- Game played with oracle.
- H is the game board.
- Player moves a coin with arrow.
- Moves: Right, Left, Cross.
- Oracle accepts/rejects moves.



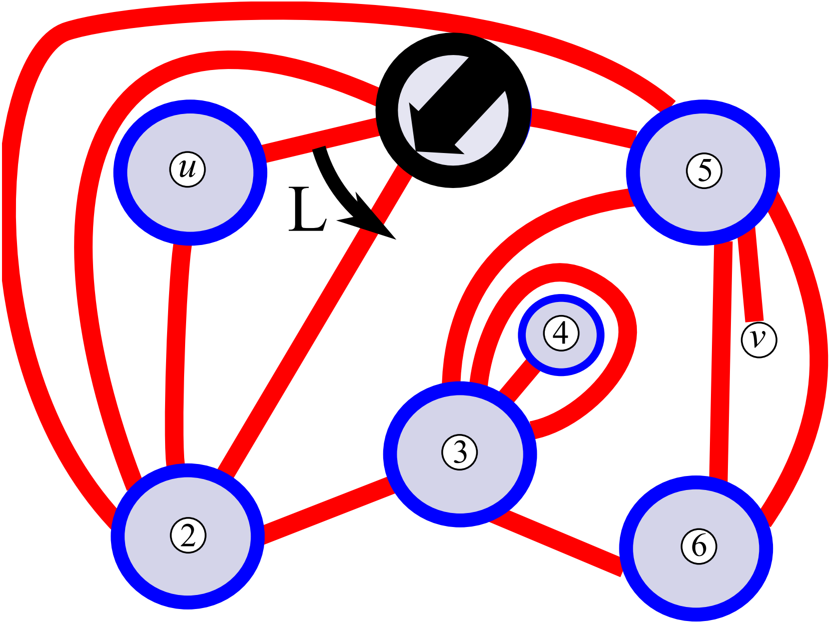
The coin.



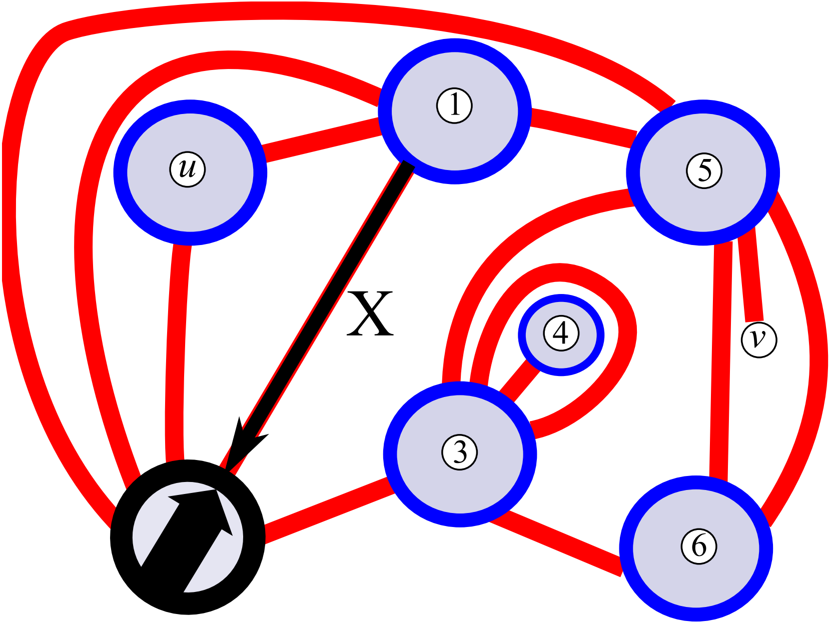
1
e_s
5
4
v
3
2
6



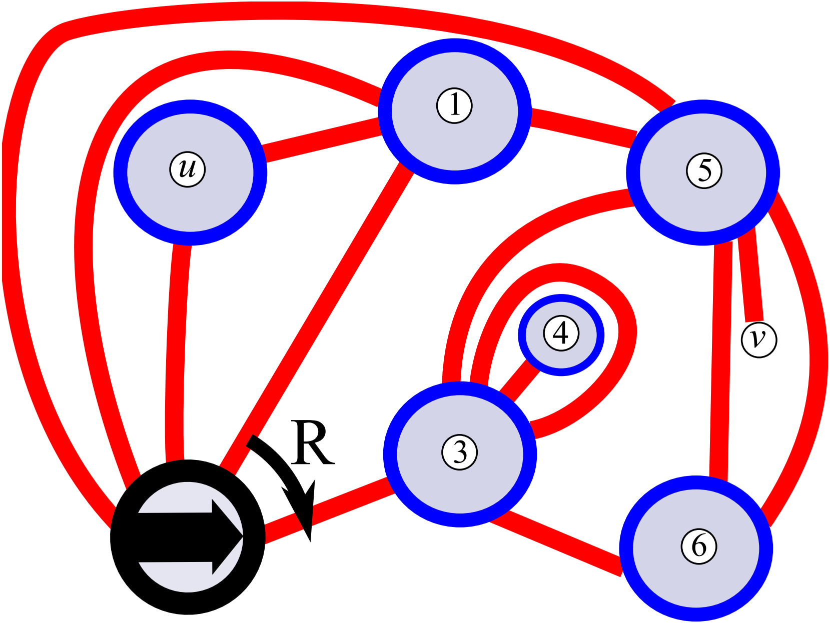
u
e_S
5
4
v
3
2
6

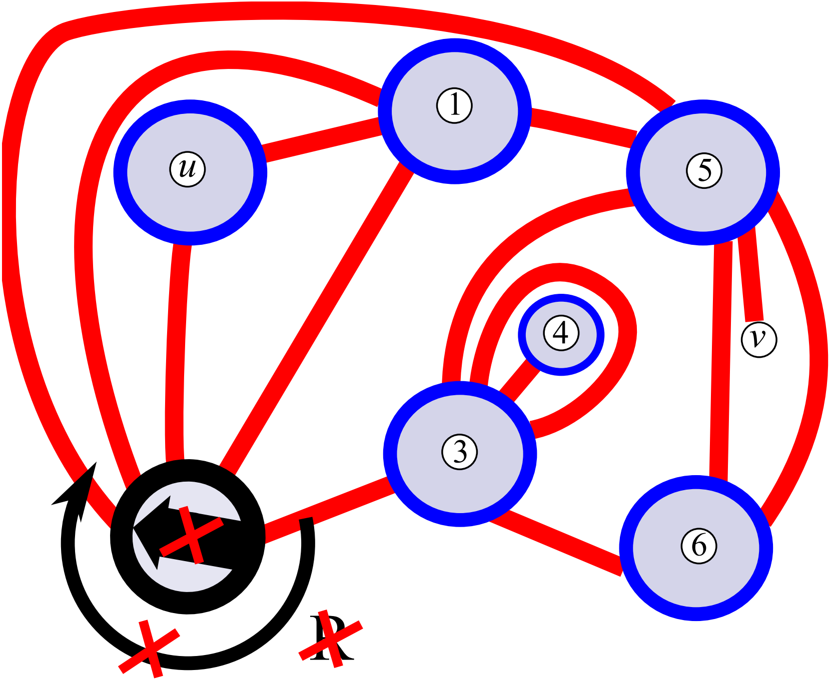


u
L
5
4
v
3
2
6

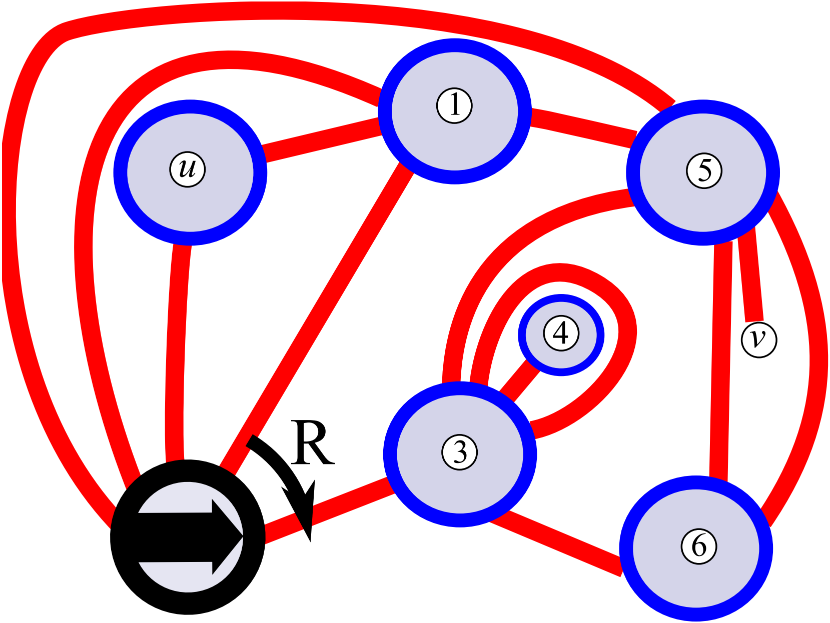


1
u
5
X
4
v
3
6

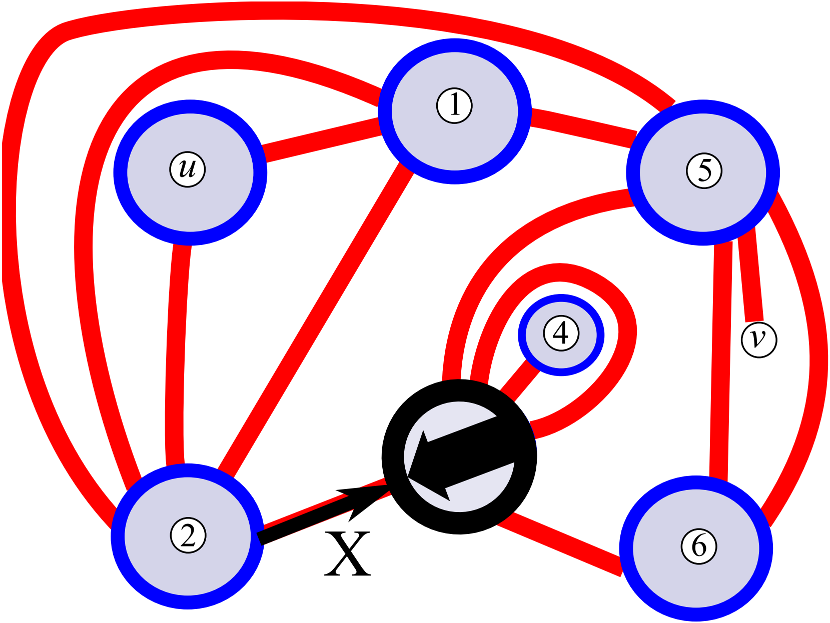

1
u
5
4
v
R
3
6



1
u
5
4
v
3
6
R

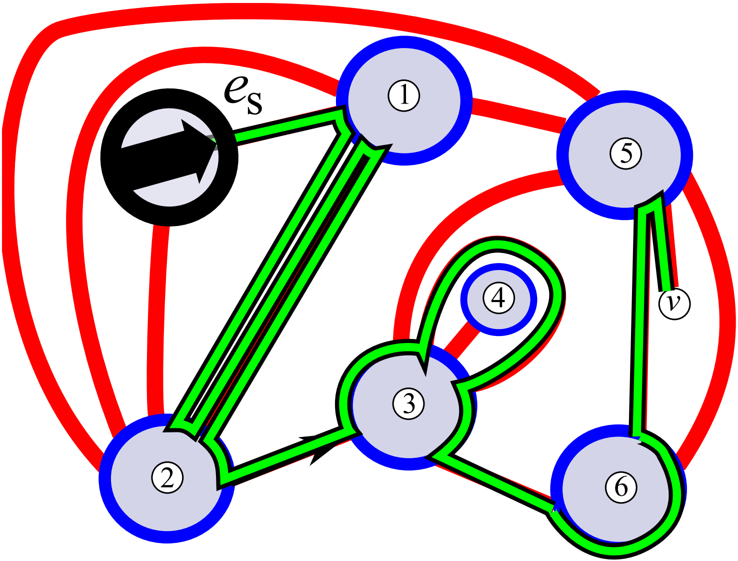


1
u
5
4
v
3
6
R


1
u
5
4
v
2
6
X

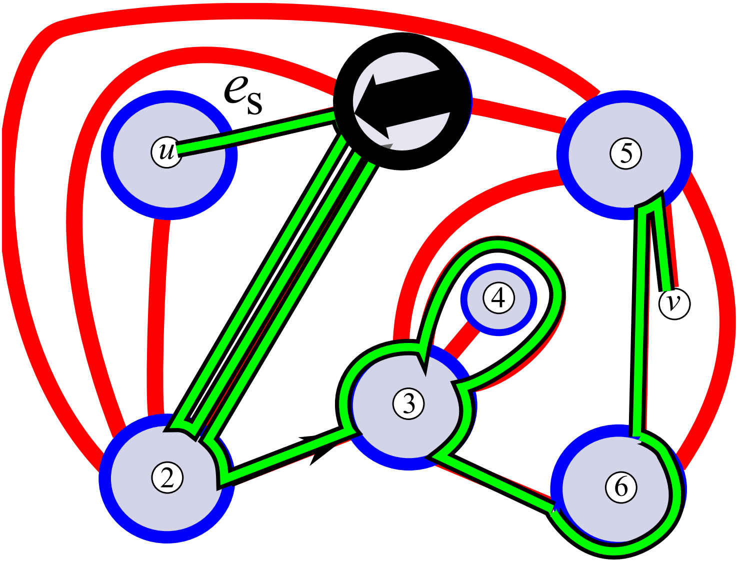
Promises

1. If you Cross, you need to rotate next.
2. If you rotate over an arc, you never need to rotate over it again (it becomes a *forbidden zone*).

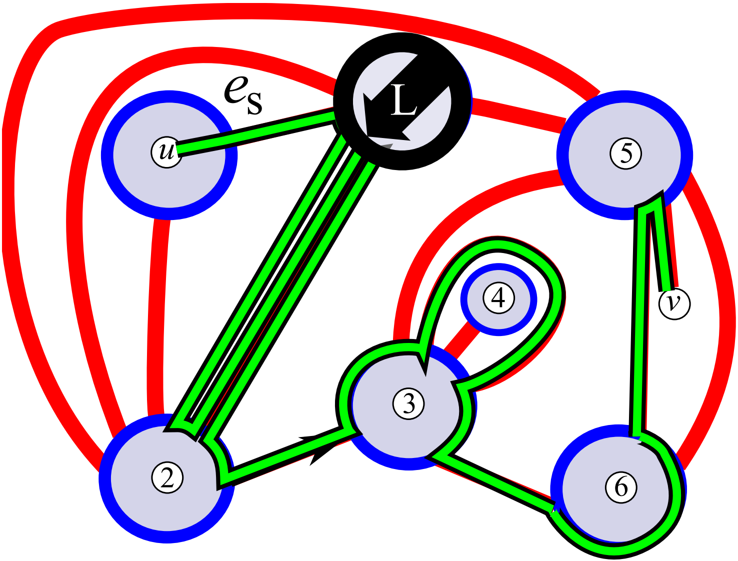


e_S
1
2
3
4
5
6
v

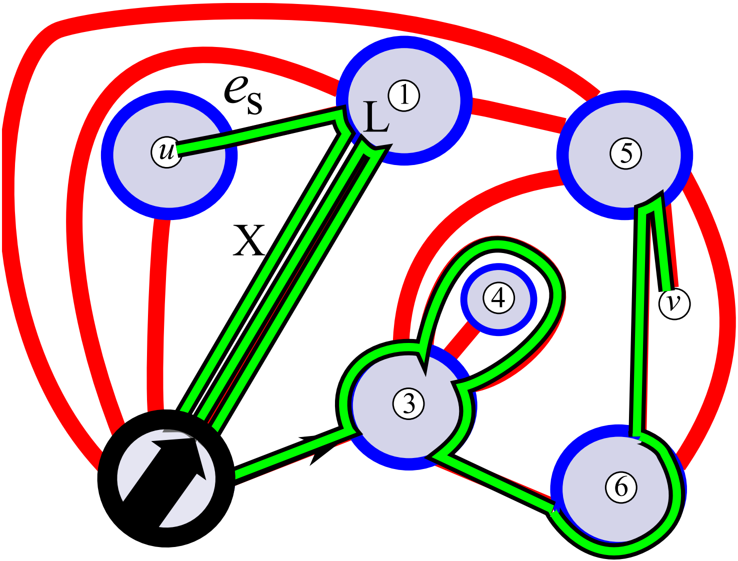
(



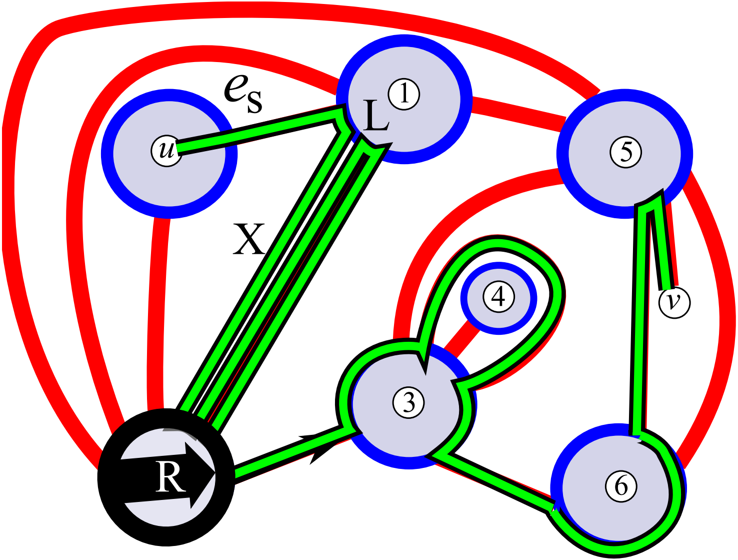
$(e_s,$



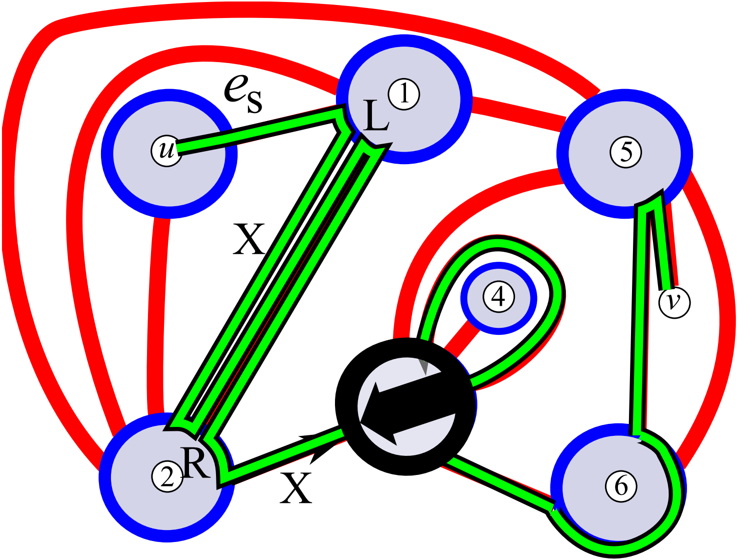
(e_s, L,



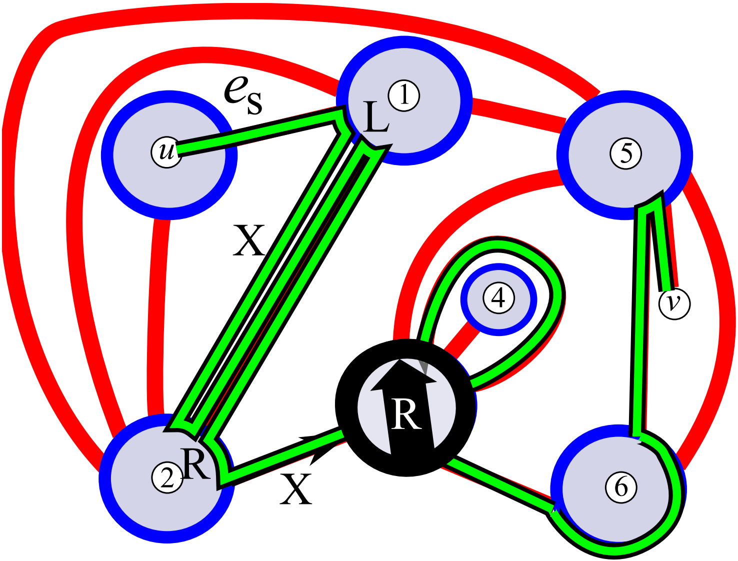
(e_s, L, X,



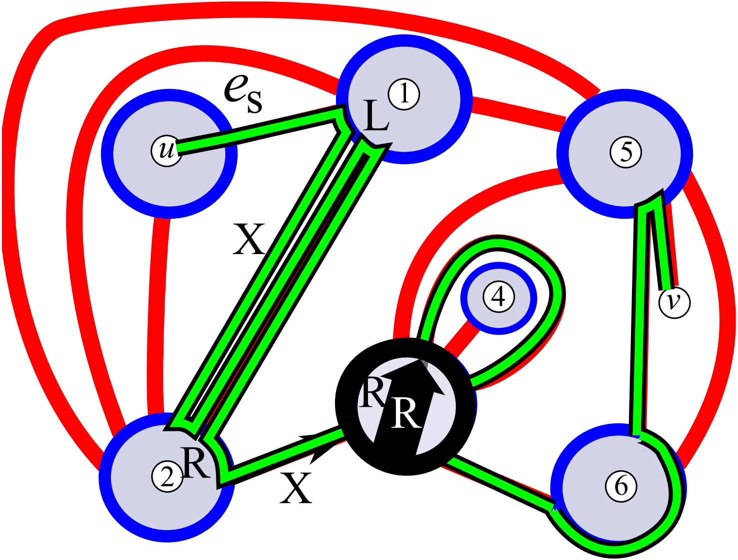
(e_s, L, X, R,



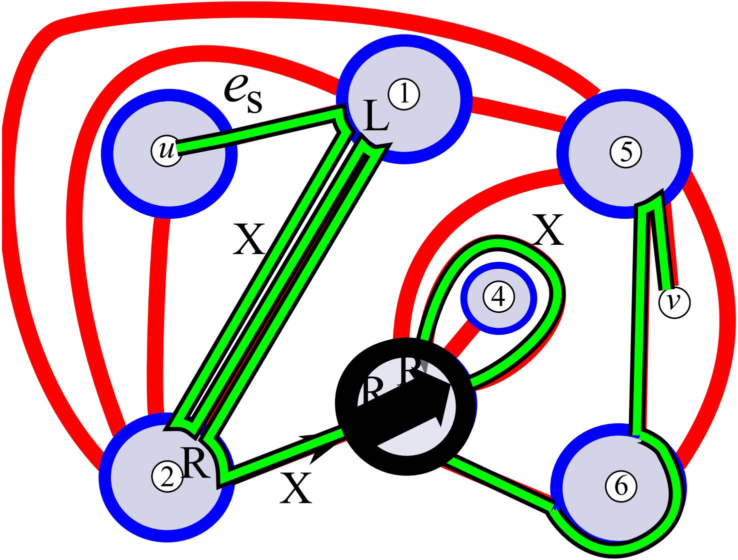
$(e_s$, L, X, R, X,



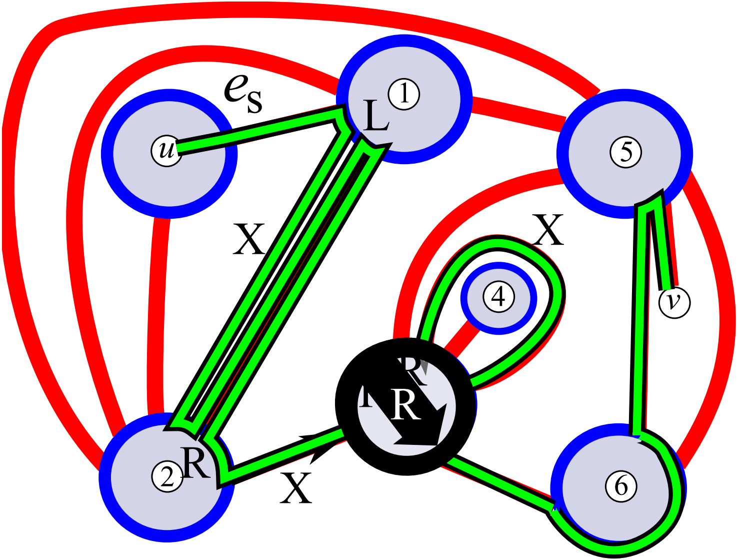
(e_s, L, X, R, X, R,



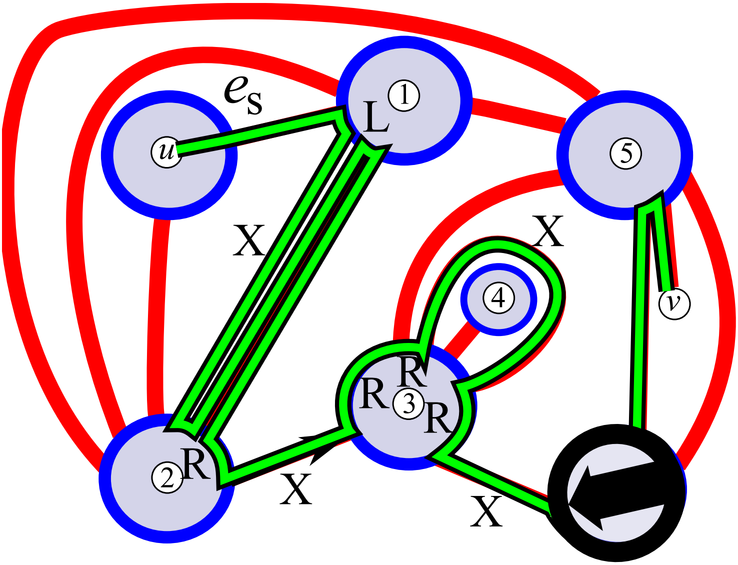
(e_s, L, X, R, X, R, R,



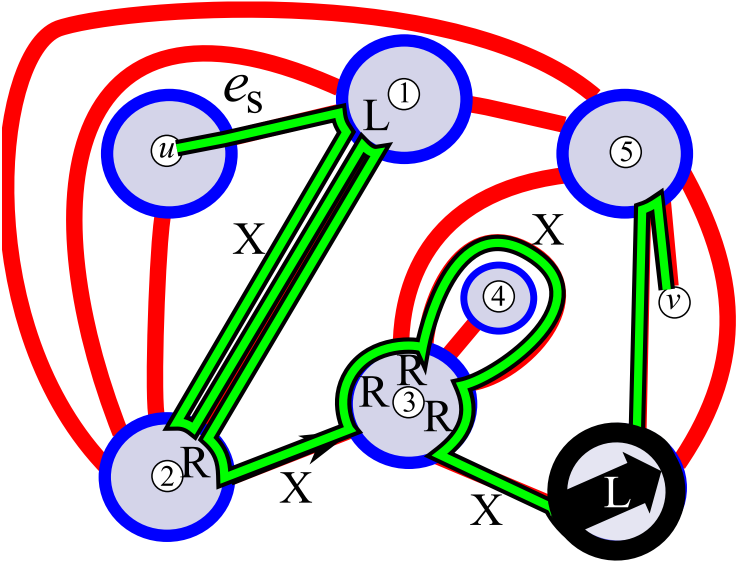
(e_s, L, X, R, X, R, R, X,



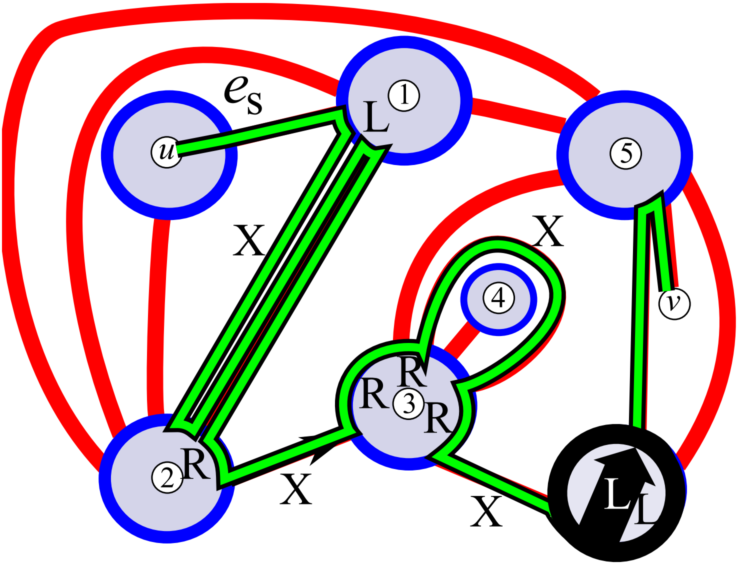
(e_s, L, X, R, X, R, R, X, R,



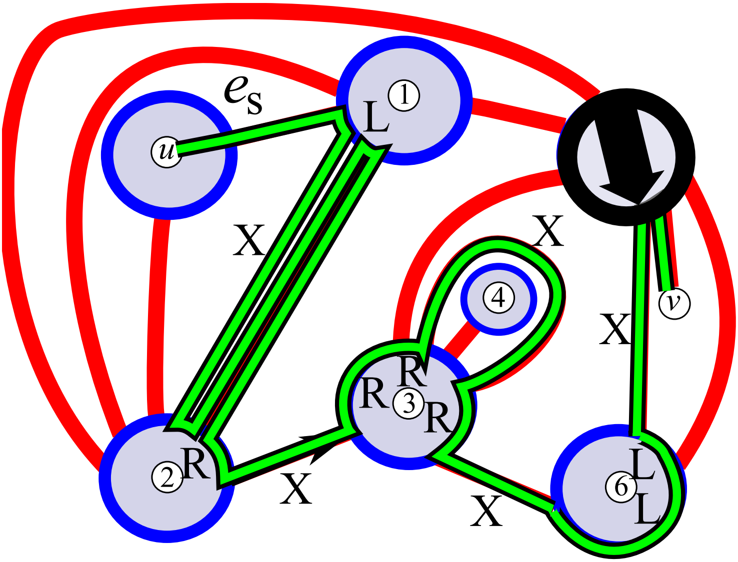
(e_s, L, X, R, X, R, R, X, R, X,



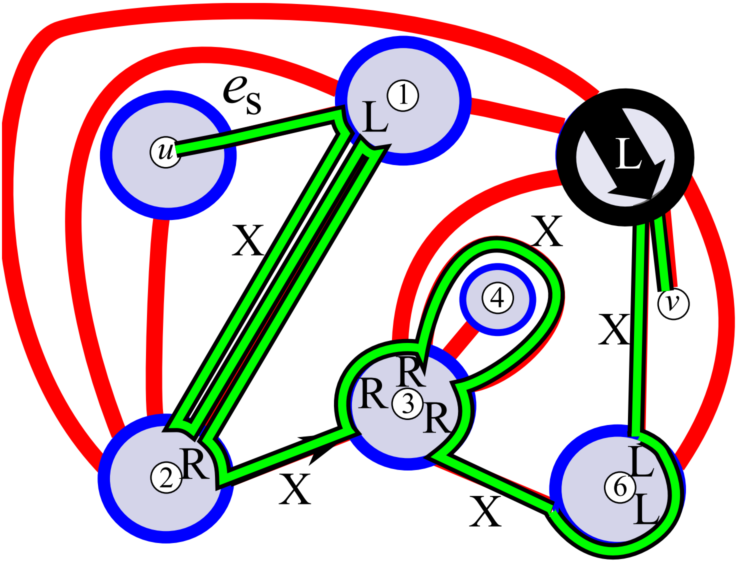
(e_s, L, X, R, X, R, R, X, R, X, L,



(e_s, L, X, R, X, R, R, X, R, X, L, L,



(e_s, L, X, R, X, R, R, X, R, X, L, L, X,



$(e_s$, L, X, R, X, R, R, X, R, X, L, L, X, L)

Proof Outline

Game to Algorithm

To convert the Coin Crawl game into algorithm, we require:

1. A log-space data structure: Explored Region.
2. Operation to detect possible moves.
3. Operation to modify region given a move.
4. Expand the region between moves.

Game to Algorithm

To convert the Coin Crawl game into algorithm, we require:

1. A log-space data structure: Explored Region. **(The coin)**
2. Operation to detect possible moves.
3. Operation to modify region given a move.
4. Expand the region between moves.

Game to Algorithm

To convert the Coin Crawl game into algorithm, we require:

1. A log-space data structure: Explored Region. **(The coin)**
2. Operation to detect possible moves. **(The oracle)**
3. Operation to modify region given a move.
4. Expand the region between moves.

Game to Algorithm

To convert the Coin Crawl game into algorithm, we require:

1. A log-space data structure: Explored Region. **(The coin)**
2. Operation to detect possible moves. **(The oracle)**
3. Operation to modify region given a move. **(A move)**
4. Expand the region between moves.

Game to Algorithm

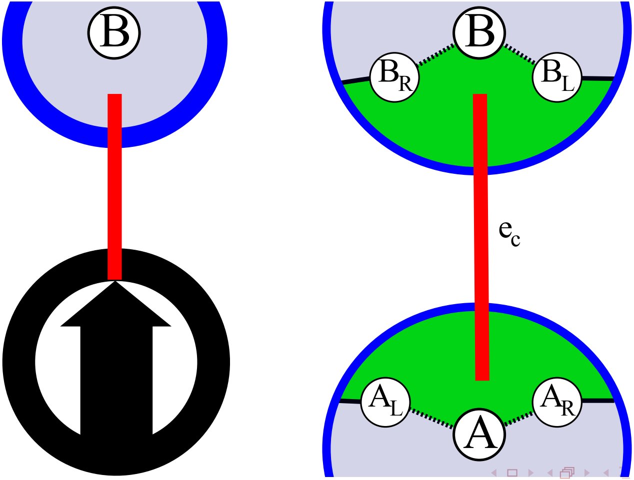
To convert the Coin Crawl game into algorithm, we require:

1. A log-space data structure: Explored Region. **(The coin)**
2. Operation to detect possible moves. **(The oracle)**
3. Operation to modify region given a move. **(A move)**
4. Expand the region between moves. **(Semi-local search)**

The Coin: Explored Region

Definition

An *explored region* is a tuple $C = (A_{\mathrm{L}}, A_{\mathrm{R}}, e_{\mathrm{c}}, B_{\mathrm{L}}, B_{\mathrm{R}})$.



The Explored Region

We need the following two properties of an explored region:

The Explored Region

We need the following two properties of an explored region:

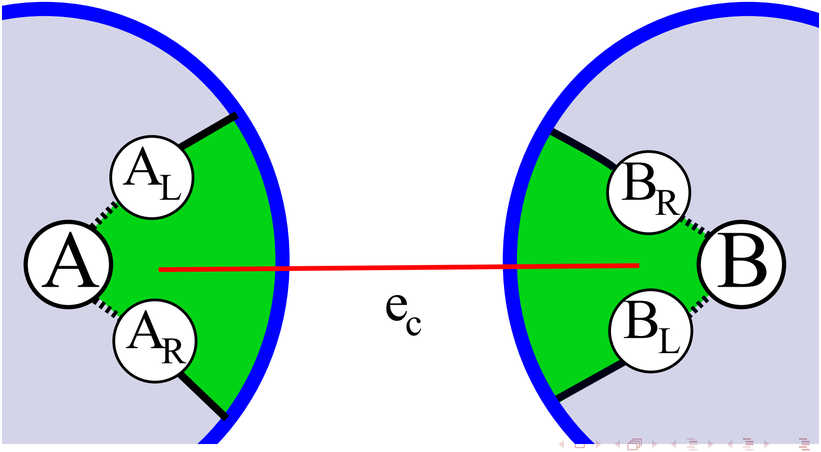
1. All launch edges with tail in the region have the head reachable.

The Explored Region

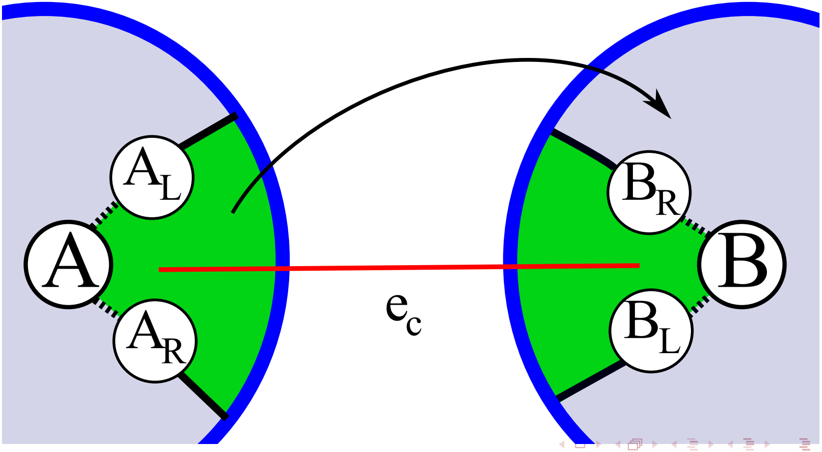
We need the following two properties of an explored region:

1. All launch edges with tail in the region have the head reachable.
2. The explored region “expands” to include launch edges reachable using tree, local, and jump edges, as well as launch edges equivalent to e_c.

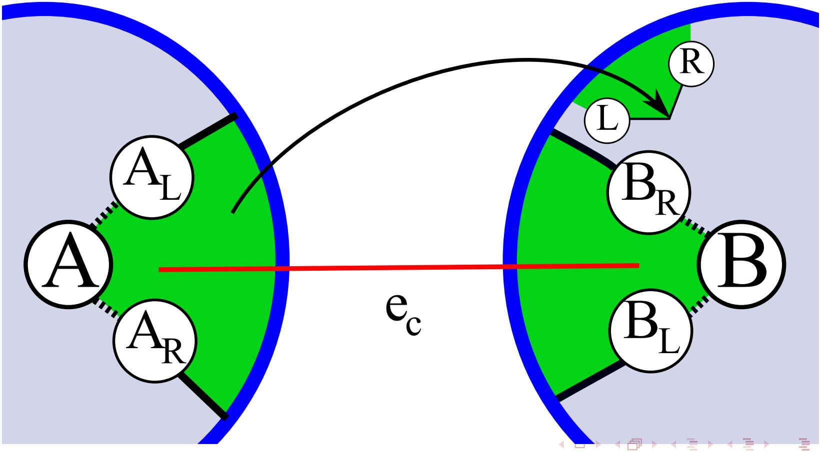
Expanding the Explored Region



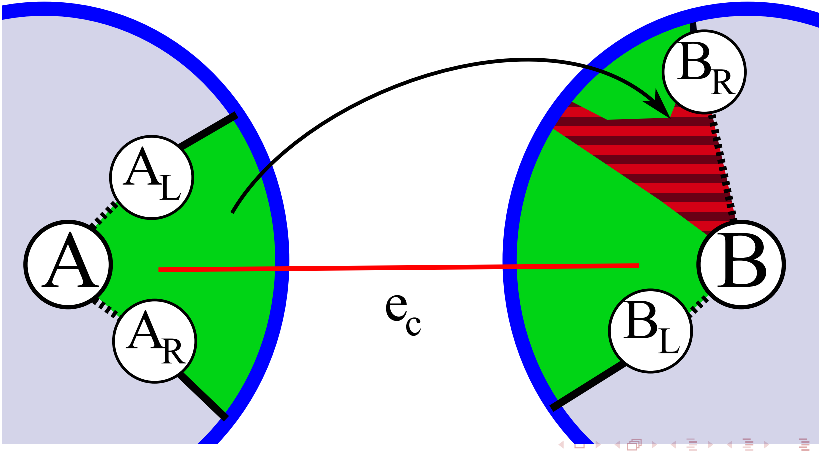
Expanding the Explored Region



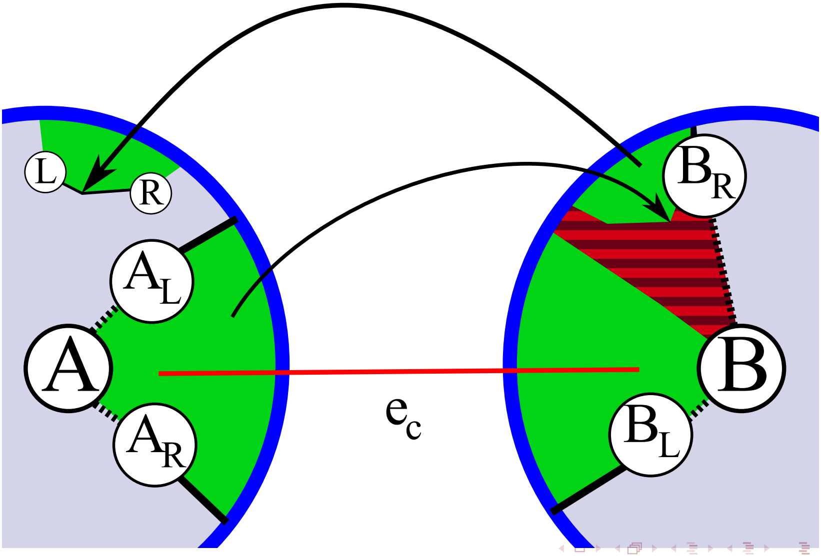
Expanding the Explored Region



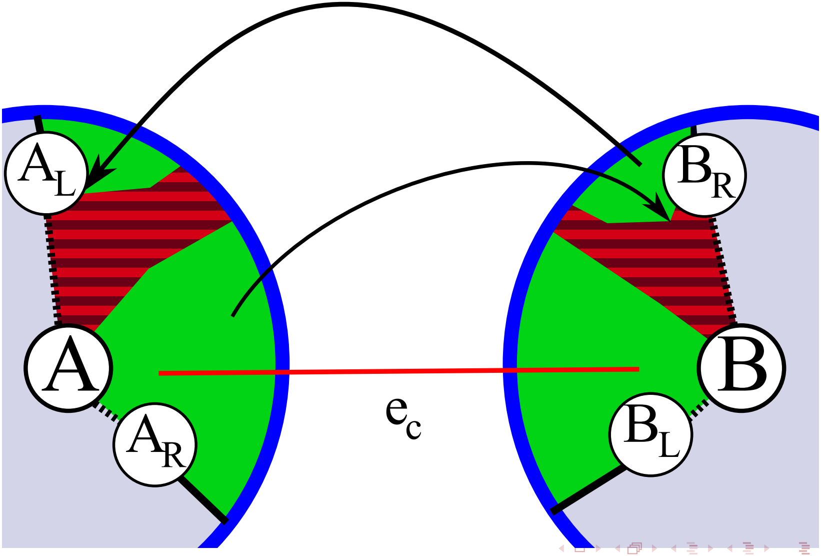
Expanding the Explored Region



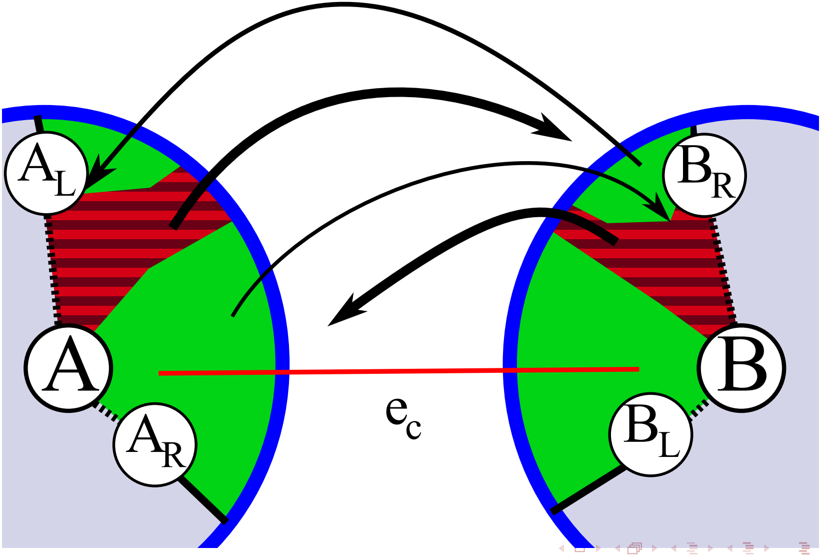
Expanding the Explored Region



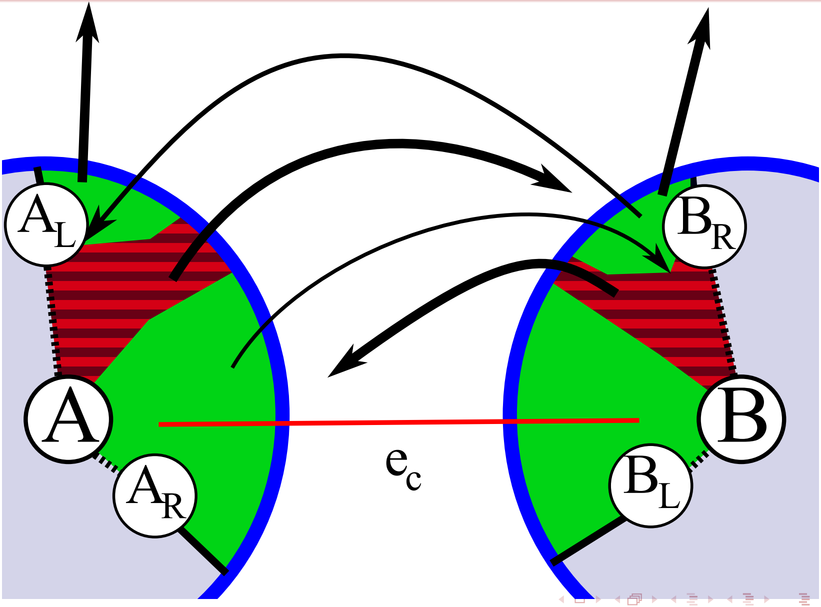
Expanding the Explored Region



Expanding the Explored Region

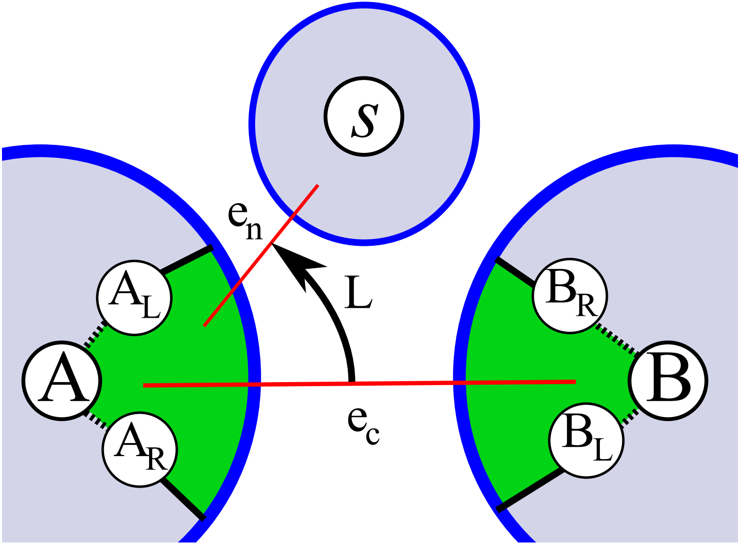


Expanding the Explored Region



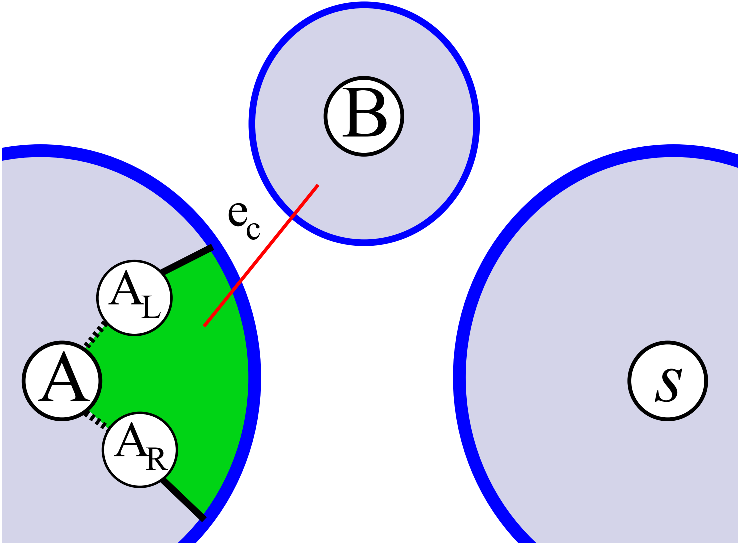
Moving the Explored Region

Rotations abandon B-side and change current edge.



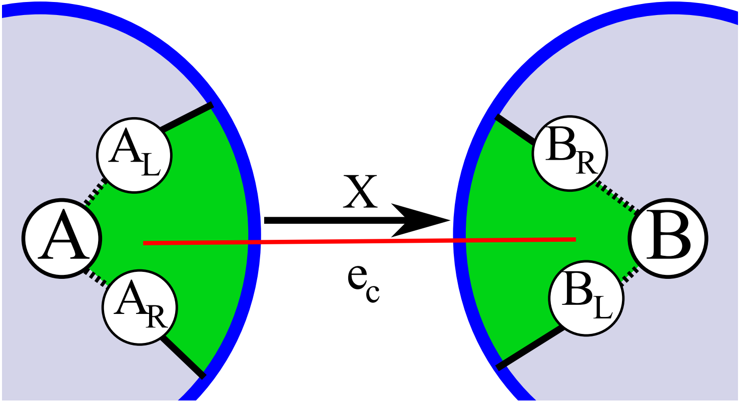
Moving the Explored Region

Rotations abandon B-side and change current edge.



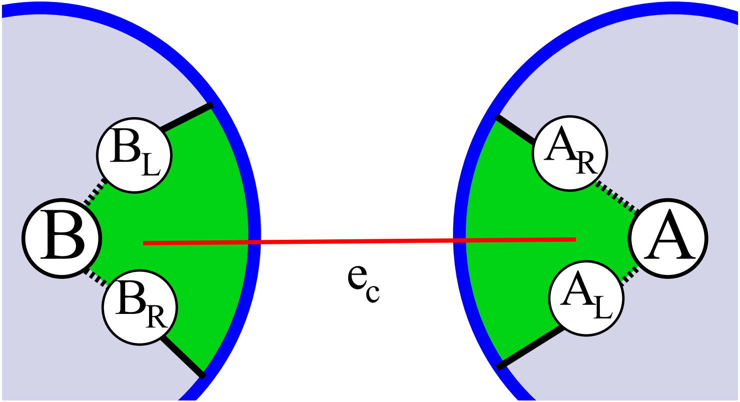
Moving the Explored Region

Cross moves swap A- and B-sides.



Moving the Explored Region

Cross moves swap A- and B-sides.



Winning the Game

If the explored region contains a launch edge to v, accept!

Proof Outline

Lemma

A “nice” path in an m-source planar DAG induces a move string of length at most $12m$.

Lemma

A "nice" path in an m-source planar DAG induces a move string of length at most $12m$.

Proof.

At most $\deg_H s_i$ rotations can occur at each source. This gives at most

$$\sum_{i=1}^{m} \deg_H s_i = 2|E(H)| \leq 6m$$

rotations. Each Cross moves precedes a rotation, at most $6m$. $\square$

Putting it Together

Theorem (Main Theorem)

The reachability problem for planar directed acyclic graphs with $m = m(n)$ sources is decidable in deterministic $O(m + \log n)$ space

Putting it Together

Theorem (Main Theorem)

The reachability problem for planar directed acyclic graphs with $m = m(n)$ sources is decidable in deterministic $O(m + \log n)$ space

Proof Idea.

Iterate over all start edges e_s and move strings σ of length $12m$. For each pair (e_s, σ), simulate the Coin Crawl game. Some pair will return successfully if and only if a $u - v$ path exists. □

A Recent Result: Background

Theorem (Savitch's Theorem: General Form)

Let A be an $s(n)$-space bounded, non-deterministic algorithm using a read-once certificate with $\ell(n)$ bits.

A can be simulated by a deterministic algorithm using $O(s(n) \log \ell(n))$ space.

A Recent Result

Theorem (Main Theorem: Alternate Form)

The reachability problem for planar directed acyclic graphs with m sources is decidable by a non-deterministic log-space algorithm using a read-once certificate of length $O(m + \log n)$.

A Recent Result

Theorem (Main Theorem: Alternate Form)

The reachability problem for planar directed acyclic graphs with m sources is decidable by a non-deterministic log-space algorithm using a read-once certificate of length $O(m + \log n)$.

Corollary

Reachability for planar DAGs with $m > \log n$ sources is decidable by a deterministic $O(\log n \cdot \log m)$-space algorithm.

A Recent Result

Theorem (Main Theorem: Alternate Form)

The reachability problem for planar directed acyclic graphs with m sources is decidable by a non-deterministic log-space algorithm using a read-once certificate of length $O(m + \log n)$.

Corollary

Reachability for planar DAGs with $m > \log n$ *sources is decidable by a deterministic* $O(\log n \cdot \log m)$*-space algorithm.*

- $m = 2^{O(\log^{\varepsilon} n)}$ *decidable in* $O(\log^{1+\varepsilon} n)$ *space.*
- $m = O(\log^{c} n)$ *decidable in* $O(\log n \log \log n)$ *space.*

A Recent Result

Theorem (Main Theorem: Alternate Form)

The reachability problem for planar directed acyclic graphs with m sources is decidable by a non-deterministic log-space algorithm using a read-once certificate of length $O(m + \log n)$.

Corollary

Reachability for planar DAGs with $m > \log n$ sources is decidable by a deterministic $O(\log n \cdot \log m)$-space algorithm.

- $m = 2^{O(\log^{\varepsilon} n)}$ *decidable in* $O(\log^{1+\varepsilon} n)$ *space.*
- $m = O(\log^{c} n)$ *decidable in* $O(\log n \log \log n)$ *space.*

For all sub-polynomial bounds on the number of sources, this result improves the best known space bound of $O(\log^2 n)$!

Future Work

1. The question: Is reachability for planar DAGs in L? What about general planar graphs?
2. An approach: Make a "smart" forest decomposition.
3. Can we utilize topological equivalence in other problems and/or surfaces?

An alternate definition of L

"L is like a graduate student: you don't have to know much, but you need to have a lot of time on your hands,"

– Jamie Radcliffe, UNL