Strategy Improvement for Parity Games

A Combinatorial Perspective

Paul Hunter

Outline

- Parity Games
- Strategy Improvement Algorithm
- Completely Unimodal Hypercubes
- Results – Old and New
- Conclusion

Parity Games

Parity Games

- Two player, zero-sum, non-cooperative, infinite game.
- Played on a finite, directed graph (V, E).
  - Bi-partite
  - Maximum out-degree 2

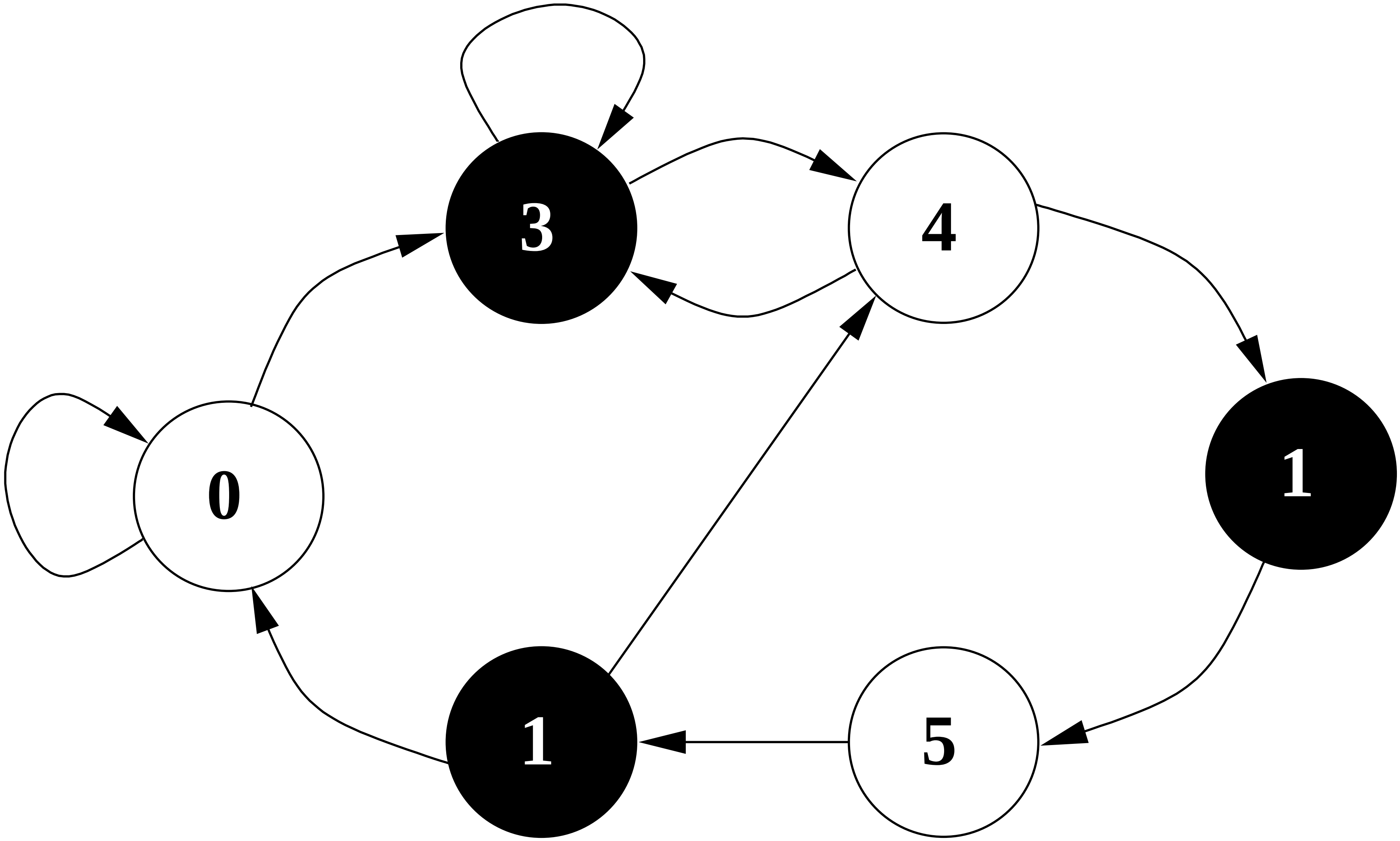
Parity Games

- Two player, zero-sum, non-cooperative, infinite game.
- Played on a finite, directed graph (V, E).
  - Bi-partite
  - Maximum out-degree 2

Players (Player 0 and Player 1) alternately move a token around the graph for an infinite number of turns, generating an infinite sequence S of vertices visited. Winner is determined by a <u>parity condition</u>:

- Priority function $\chi : V \to \mathbb{P} \qquad (\mathbb{P} \leq \omega)$
- Player 0 wins if and only if $\max_{v \in S} \chi(v)$ is even.

Parity Games – Example



Parity Games – Facts

- Determined – from any vertex one player has a strategy to defeat any play by the other player

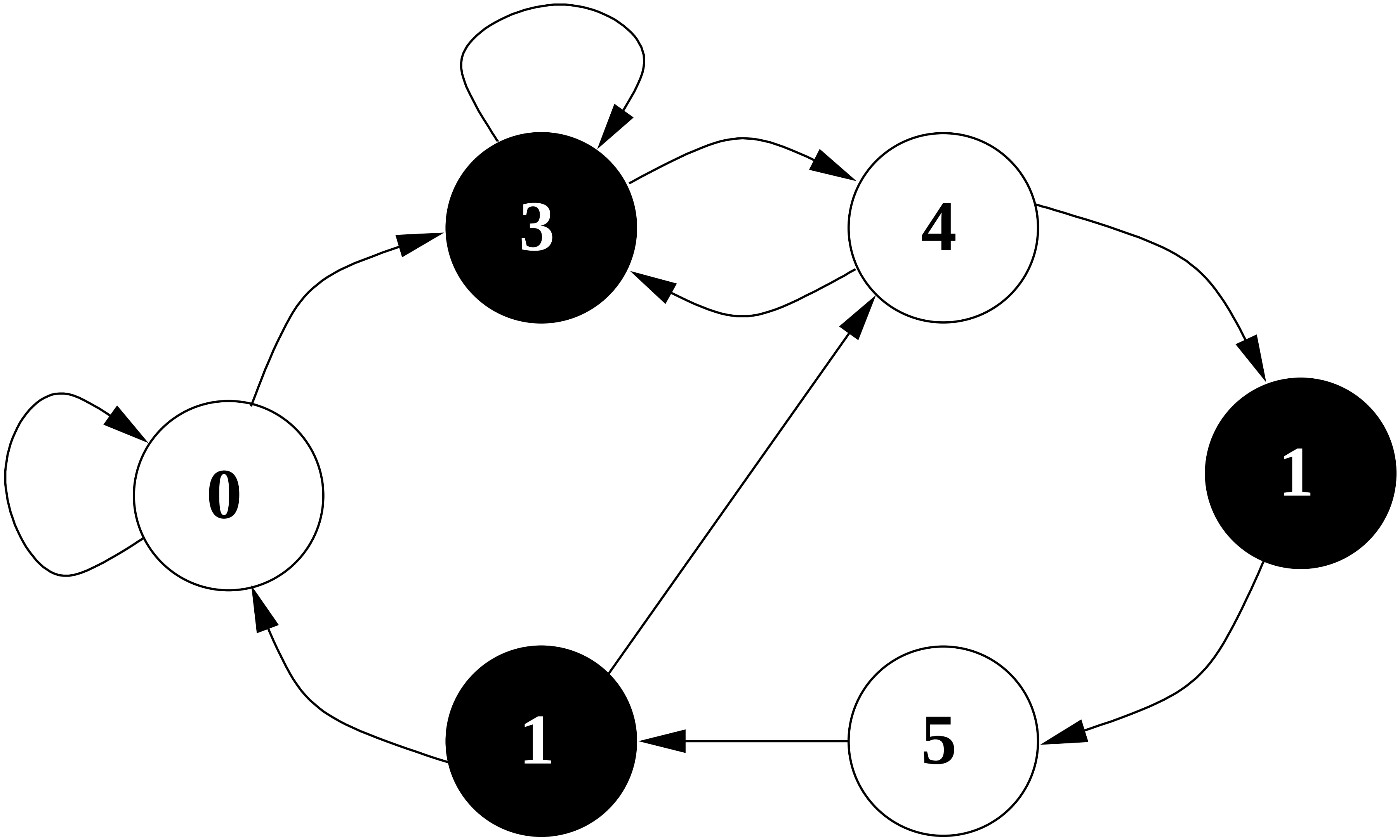
Parity Games – Facts

- Determined – from any vertex one player has a strategy to defeat any play by the other player
- Polynomially equivalent to μ-calculus model checking

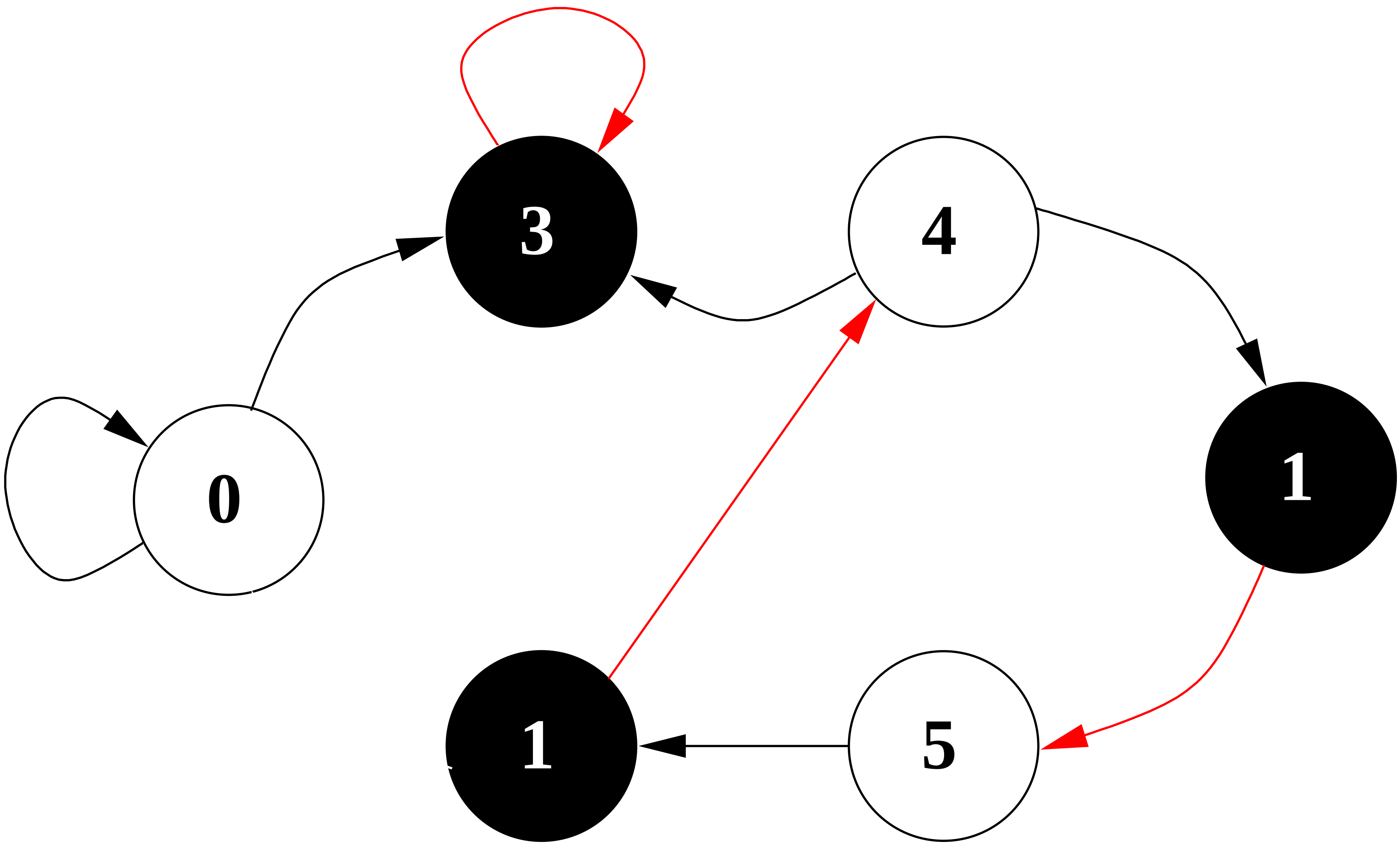
Parity Games – Facts

- Determined – from any vertex one player has a strategy to defeat any play by the other player
- Polynomially equivalent to μ-calculus model checking
- Whichever player has a winning strategy has a positional (memoryless) winning strategy

Parity Games – Winning strategy



Parity Games – Winning strategy



Parity Games – Complexity

Memoryless strategies imply that deciding Parity games is in NP $\cap$ co-NP.

Open problem: *Is deciding Parity games in P?*

Best known algorithm (Jurdziński 2000)

$$O\left(d|E|\left(\frac{|V|}{\lfloor d/2 \rfloor}\right)^{\lfloor d/2 \rfloor}\right)$$

where d is the number of priorities.

Recent approach is <u>strategy improvement</u>.

Strategy Improvement

Strategy Improvement

Introduced by Vöge and Jurdziński, 2000.

Works by "improving" memoryless strategies until optimum is reached.

Naïve time complexity analysis gives $O(|V||E|2^{|V|})$ upper bound, but no known example worse than linear time has been found!

Strategy Improvement

Introduced by Vöge and Jurdziński, 2000.

Works by "improving" memoryless strategies until optimum is reached.

Naïve time complexity analysis gives $O(|V||E|2^{|V|})$ upper bound, but no known example worse than linear time has been found!

Question: *What is the exact complexity of this algorithm?*

Strategy Improvement – Valuations

A valuation is a function

$$\varphi : V \to \mathbb{P} \times \mathcal{P}(\mathbb{P}) \times \omega$$

which assigns to each vertex:

- A loop priority
- A set of priorities, and
- A natural number

Intuitively, a valuation corresponds to a “best-play” counter-strategy.

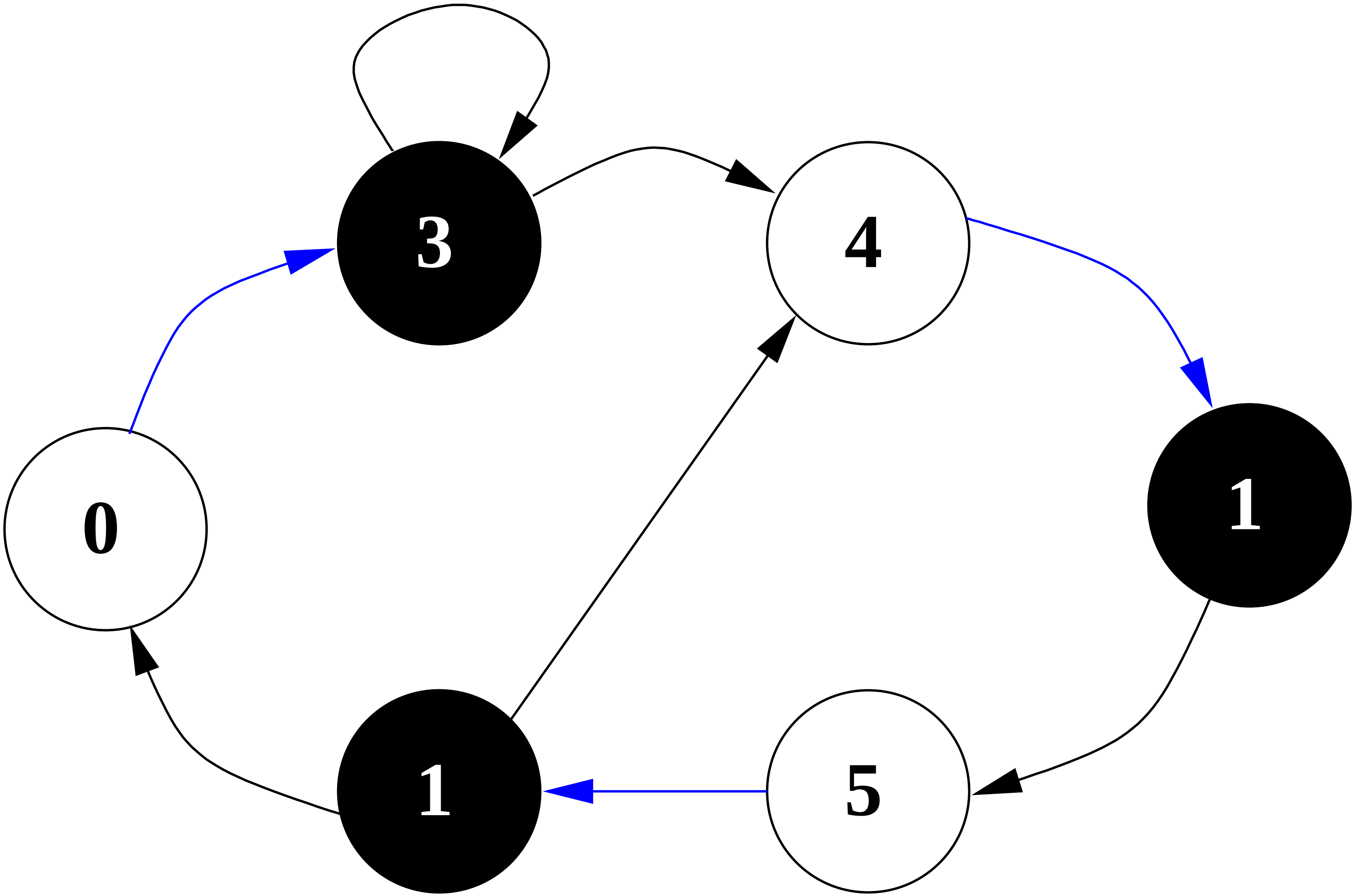
Strategy Improvement – Valuations

We can partially order valuations lexicographically according to what is best for Player 1

- High even priorities $\preceq$ Low even $\preceq$ Low odd $\preceq$ High odd
- For sets P and Q, $P \prec Q$ if $\max(P\Delta Q)$ is odd and in Q or even and in P
- Path lengths depend on the loop priority – short paths are better if the loop priority is odd

A $\preceq$-maximal valuation is 1-optimal.

Valuation example



Strategy Improvement – Algorithm

- Choose a memoryless strategy σ for Player 0

Strategy Improvement – Algorithm

- Choose a memoryless strategy σ for Player 0
- Compute a 1-optimal valuation φ

Strategy Improvement – Algorithm

- Choose a memoryless strategy σ for Player 0
- Compute a 1-optimal valuation φ
- For each $x \in V$ where Player 0 has a choice:
  - Let y be the successor of x which is not $\sigma(x)$
  - If $\varphi(y) \prec \varphi(\sigma(x))$ change σ to $\sigma' = \sigma[x \mapsto y]$.

Strategy Improvement – Algorithm

- Choose a memoryless strategy σ for Player 0
- Compute a 1-optimal valuation φ
- For each $x \in V$ where Player 0 has a choice:
  - Let y be the successor of x which is not $\sigma(x)$
  - If $\varphi(y) \prec \varphi(\sigma(x))$ change σ to $\sigma' = \sigma[x \mapsto y]$.
- Return to step 2 until no changes are made.

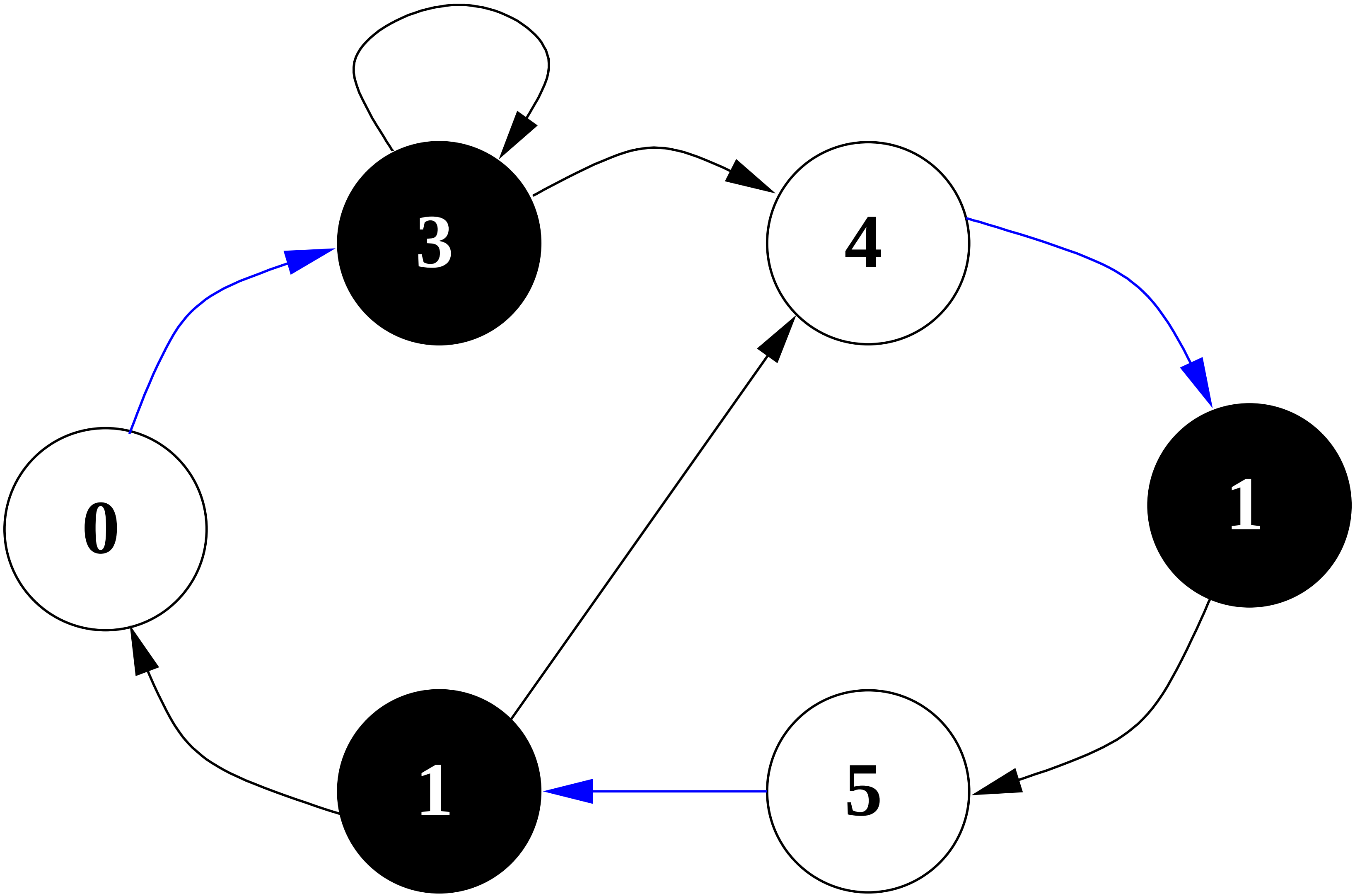
Strategy Improvement – Algorithm

- Choose a memoryless strategy σ for Player 0
- Compute a 1-optimal valuation φ
- For each $x \in V$ where Player 0 has a choice:
  - Let y be the successor of x which is not $\sigma(x)$
  - If $\varphi(y) \prec \varphi(\sigma(x))$ change σ to $\sigma' = \sigma[x \mapsto y]$.
- Return to step 2 until no changes are made.

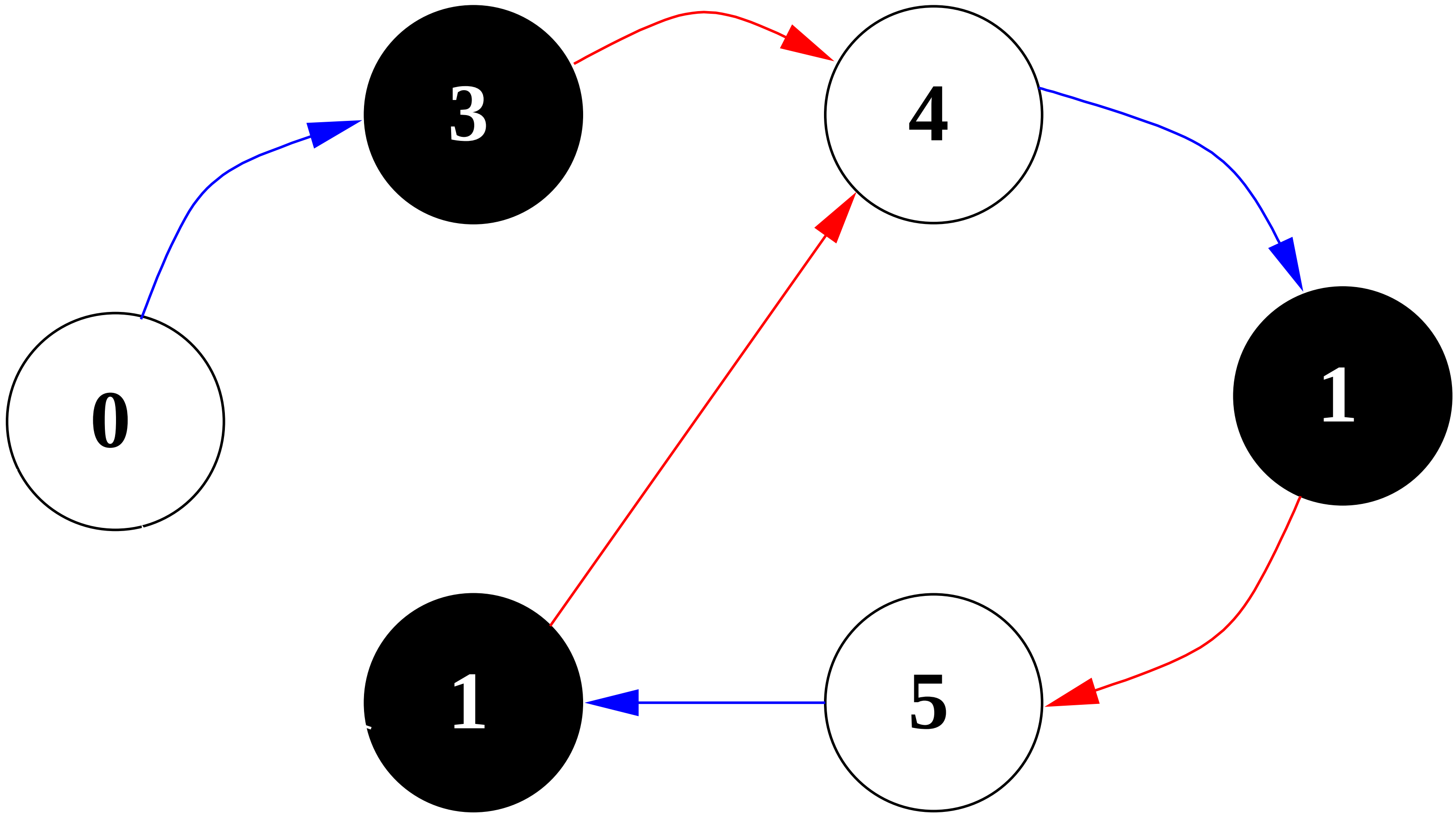
At this point σ is the best Player 0 can do, so it is straightforward to determine each player's winning sets.

Note that we are changing the strategy at different vertices simultaneously.

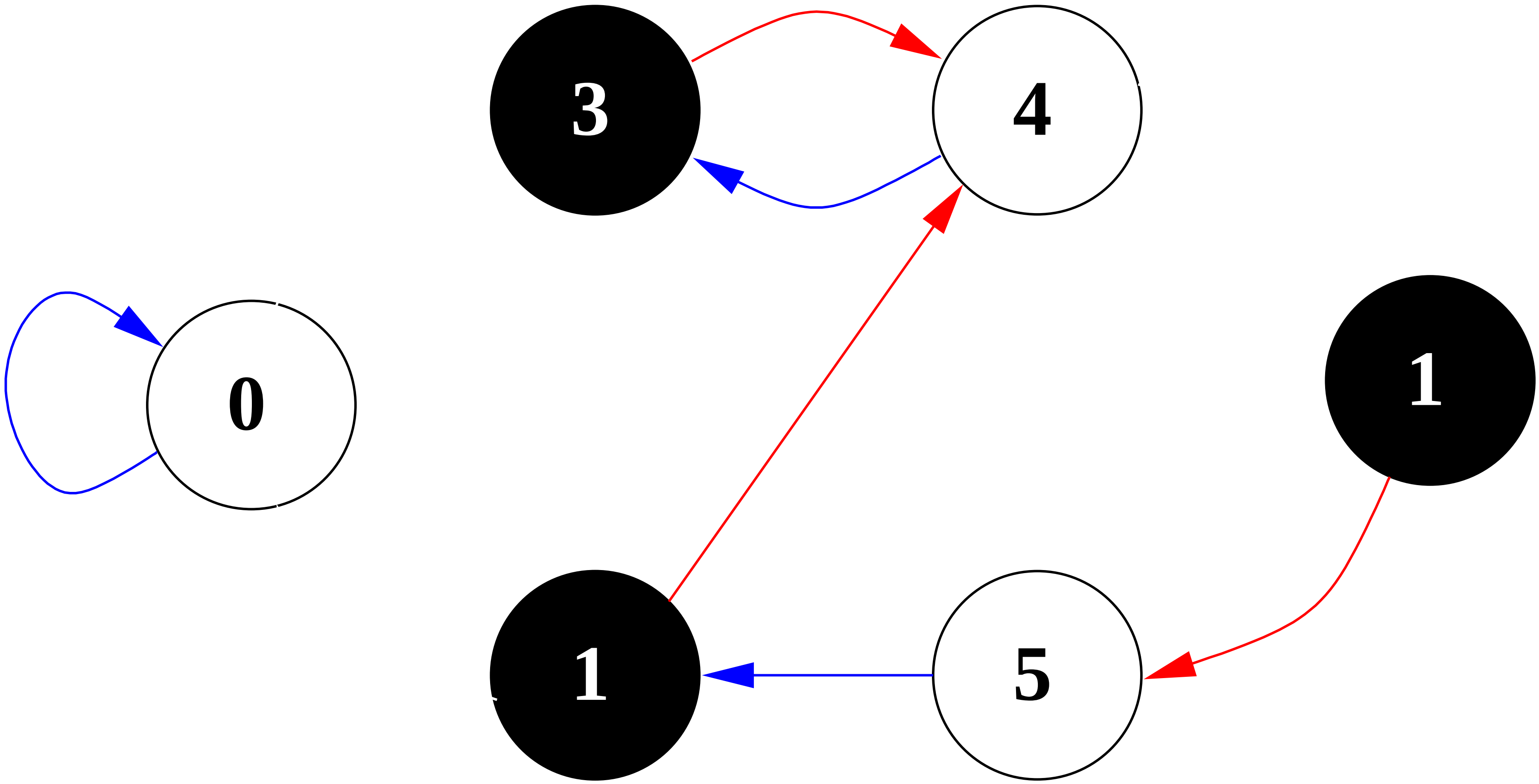
Strategy Improvement – Example



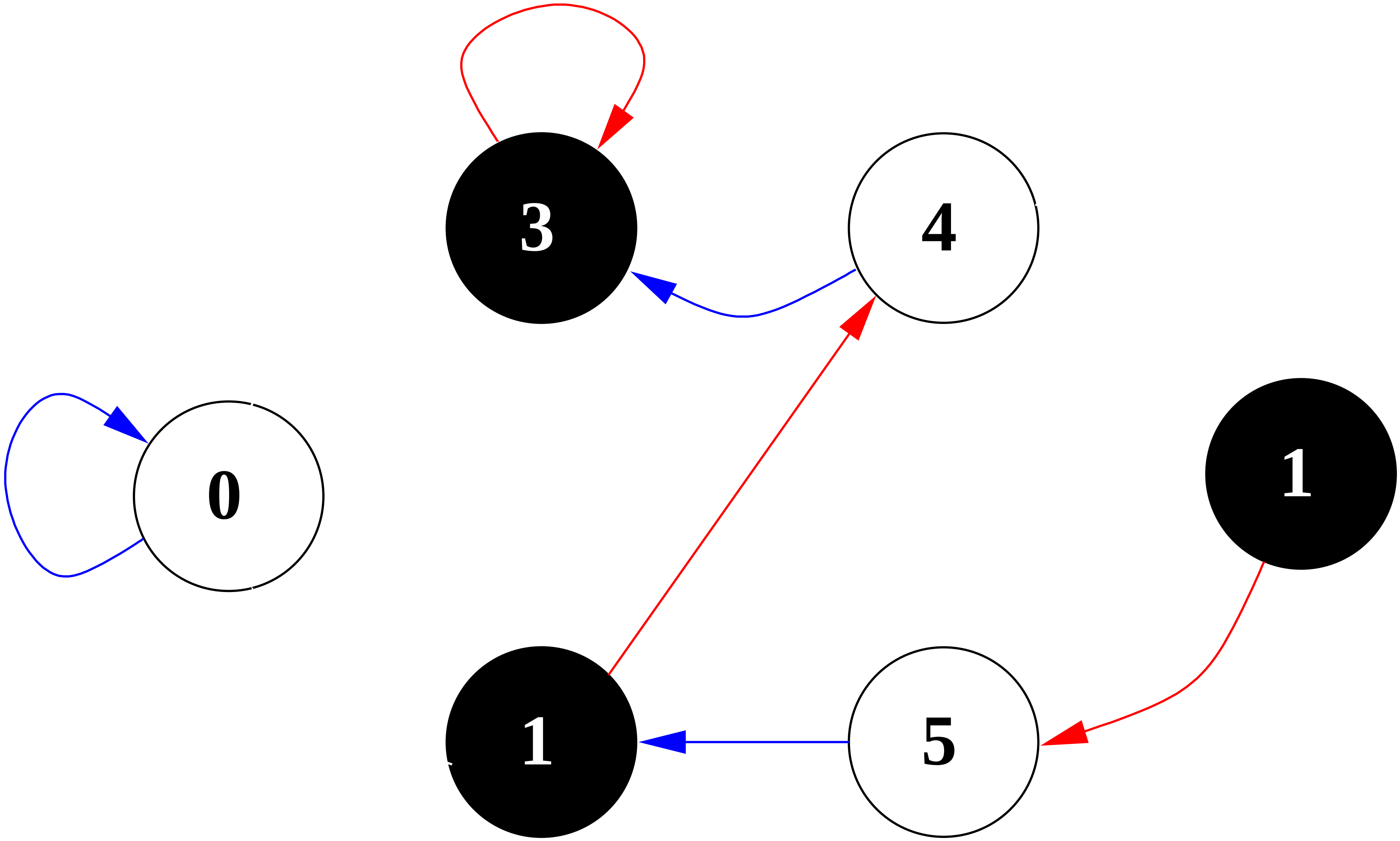
Strategy Improvement – Example



Strategy Improvement – Example



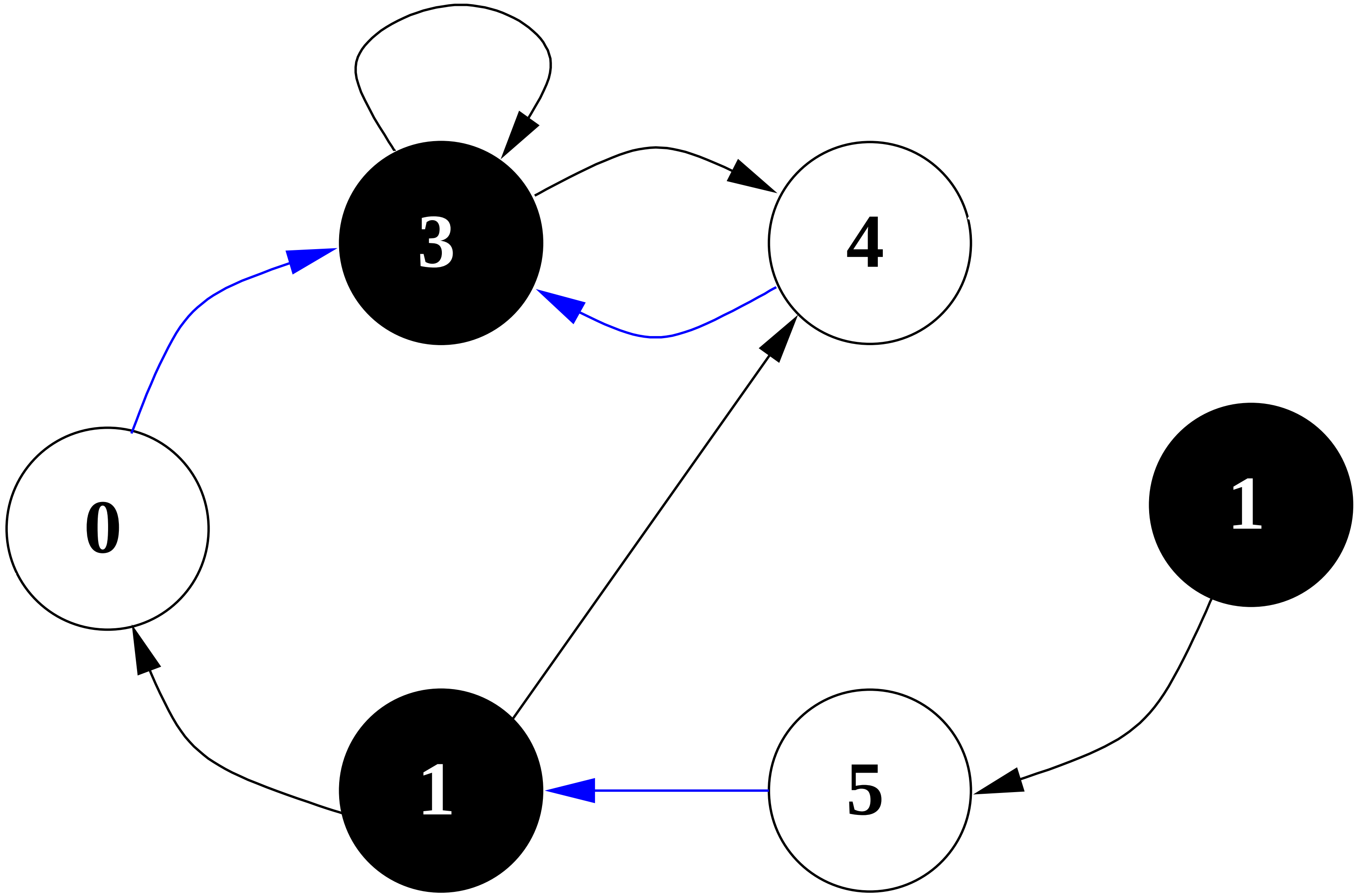
Strategy Improvement – Example



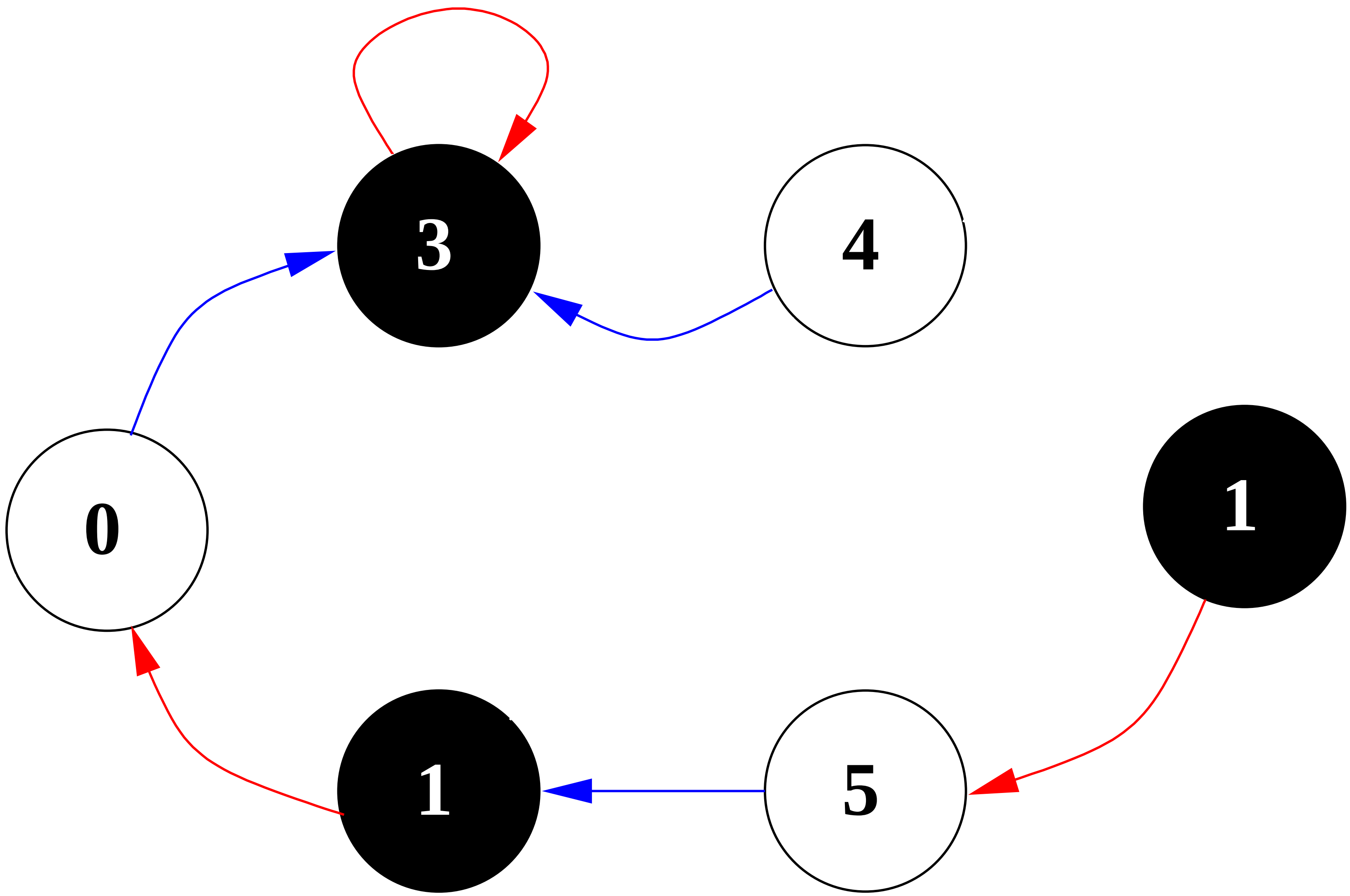
Strategy Improvement – Comments

Inherent asymmetry in algorithm. We can extract a strategy from a valuation, so why not compute a 0-optimal valuation and use this to improve σ?

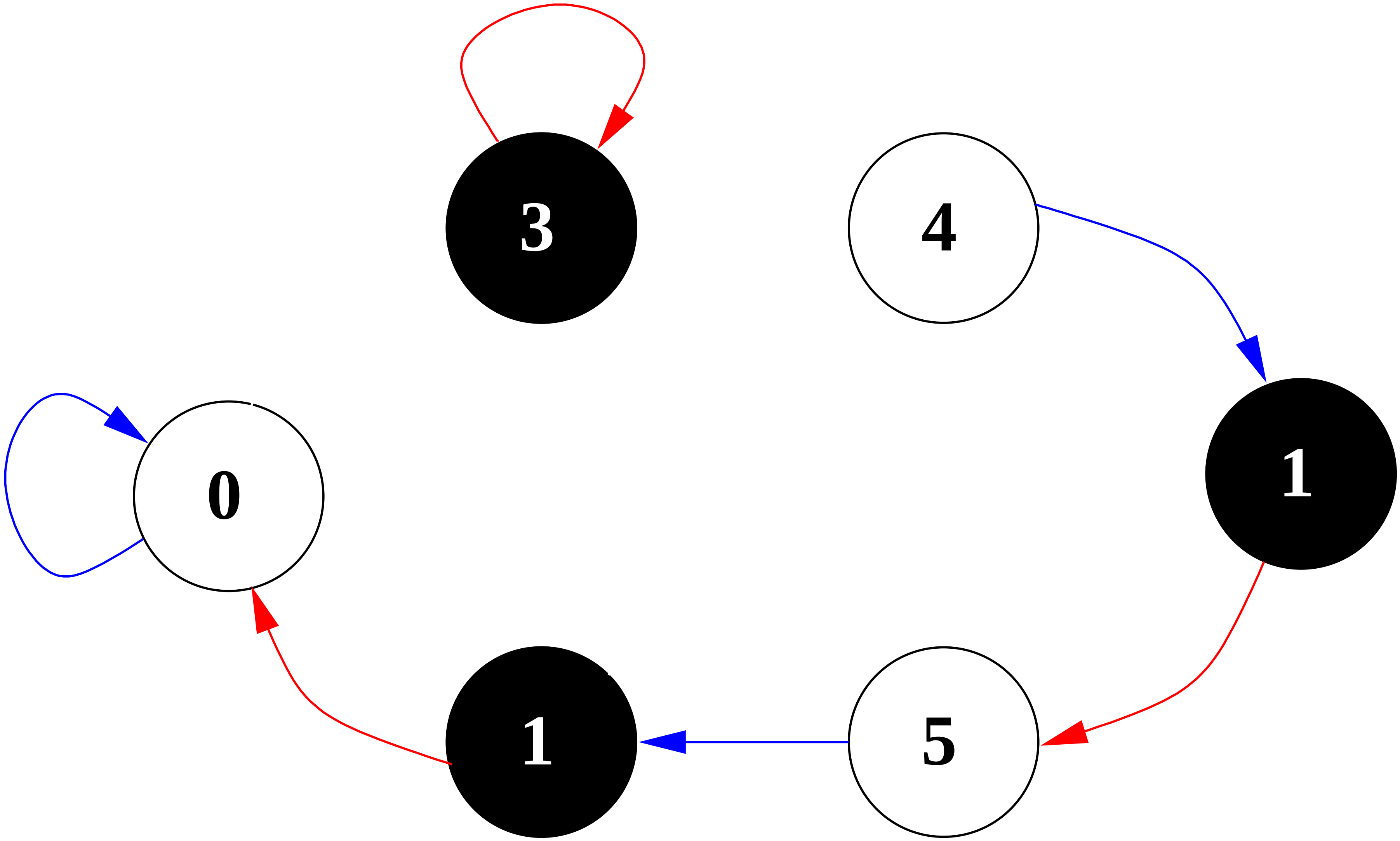
Strategy Improvement – Asymmetry



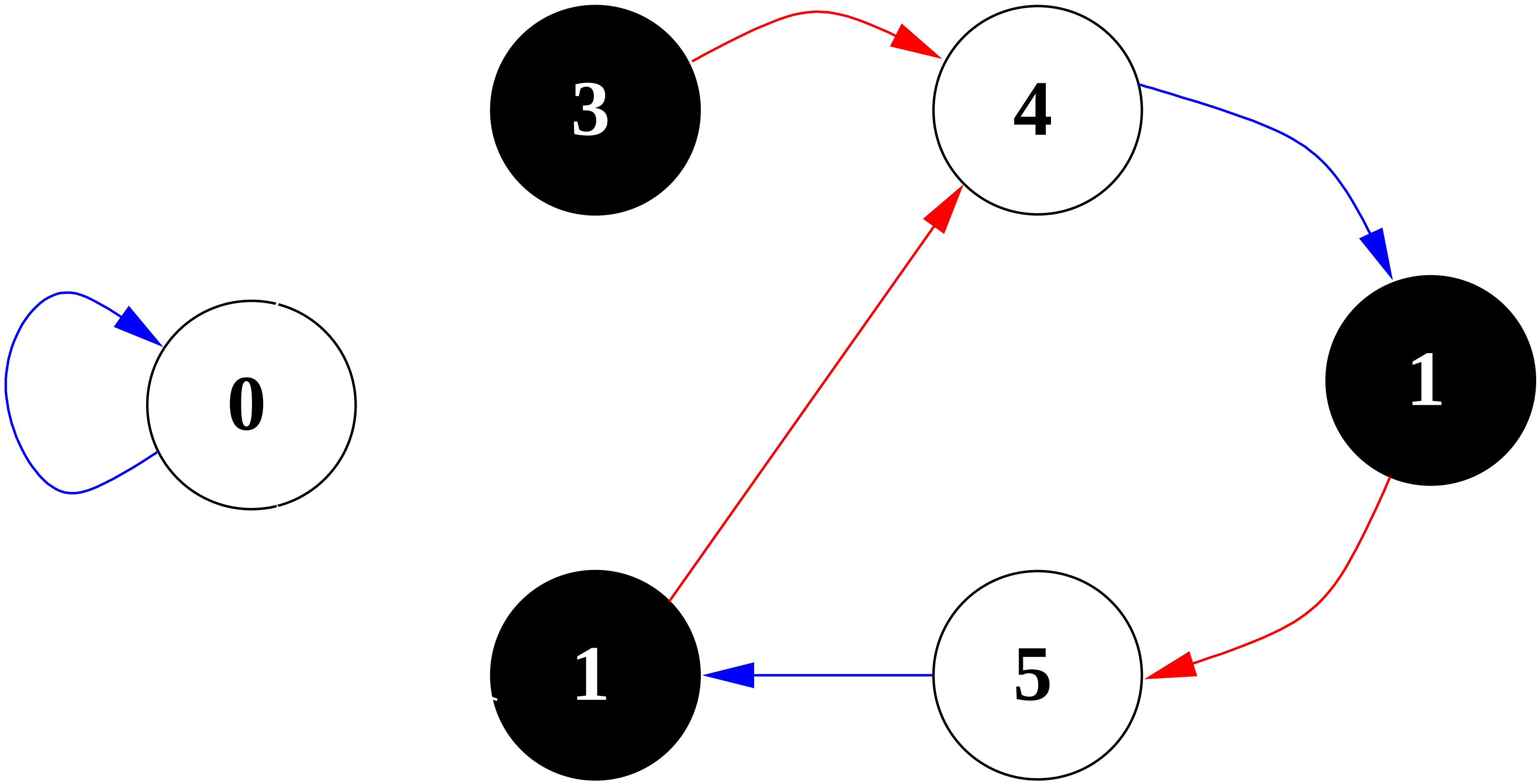
Strategy Improvement – Asymmetry



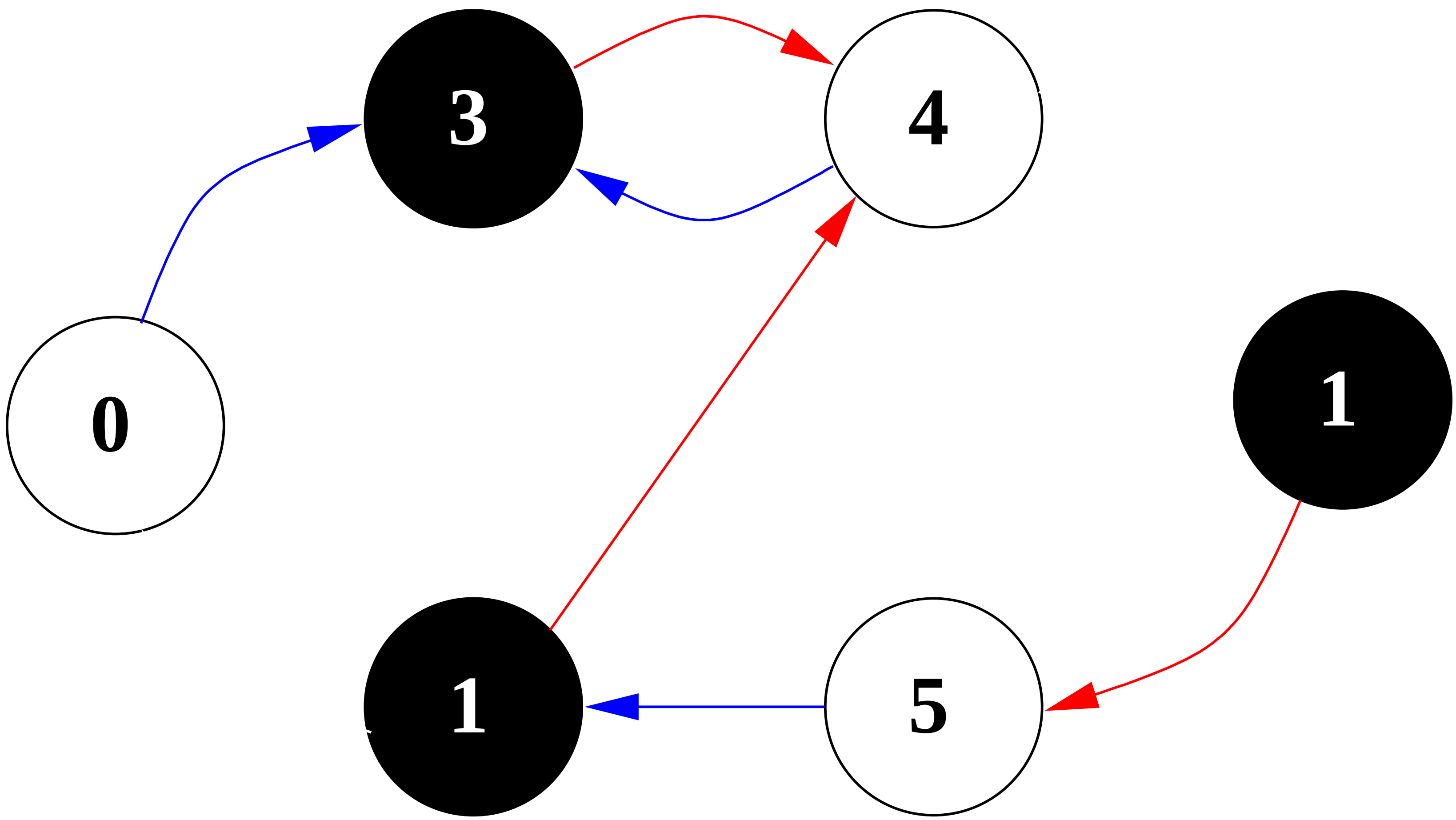
Strategy Improvement – Asymmetry



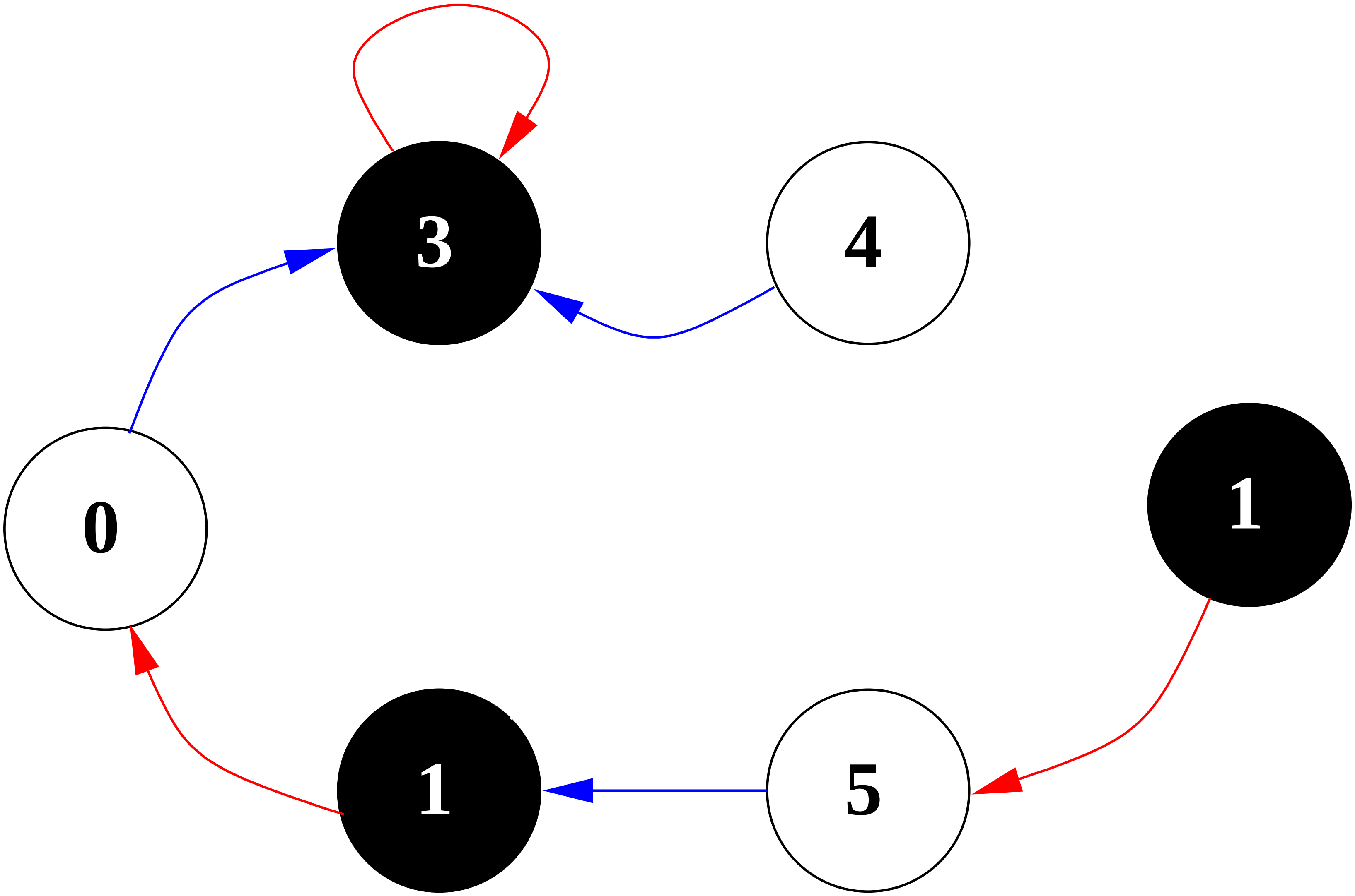
Strategy Improvement – Asymmetry



Strategy Improvement – Asymmetry



Strategy Improvement – Asymmetry



Completely Unimodal Hypercubes

Completely Unimodal Hypercubes

A psuedo-boolean function (PBF) of dimension n is a function from the n-dimensional boolean hypercube $\{0,1\}^n$ to ω.

Standard problem: *Find a local/global minimum/maximum*

This problem motivated the Polynomial Local Search (PLS) complexity class.

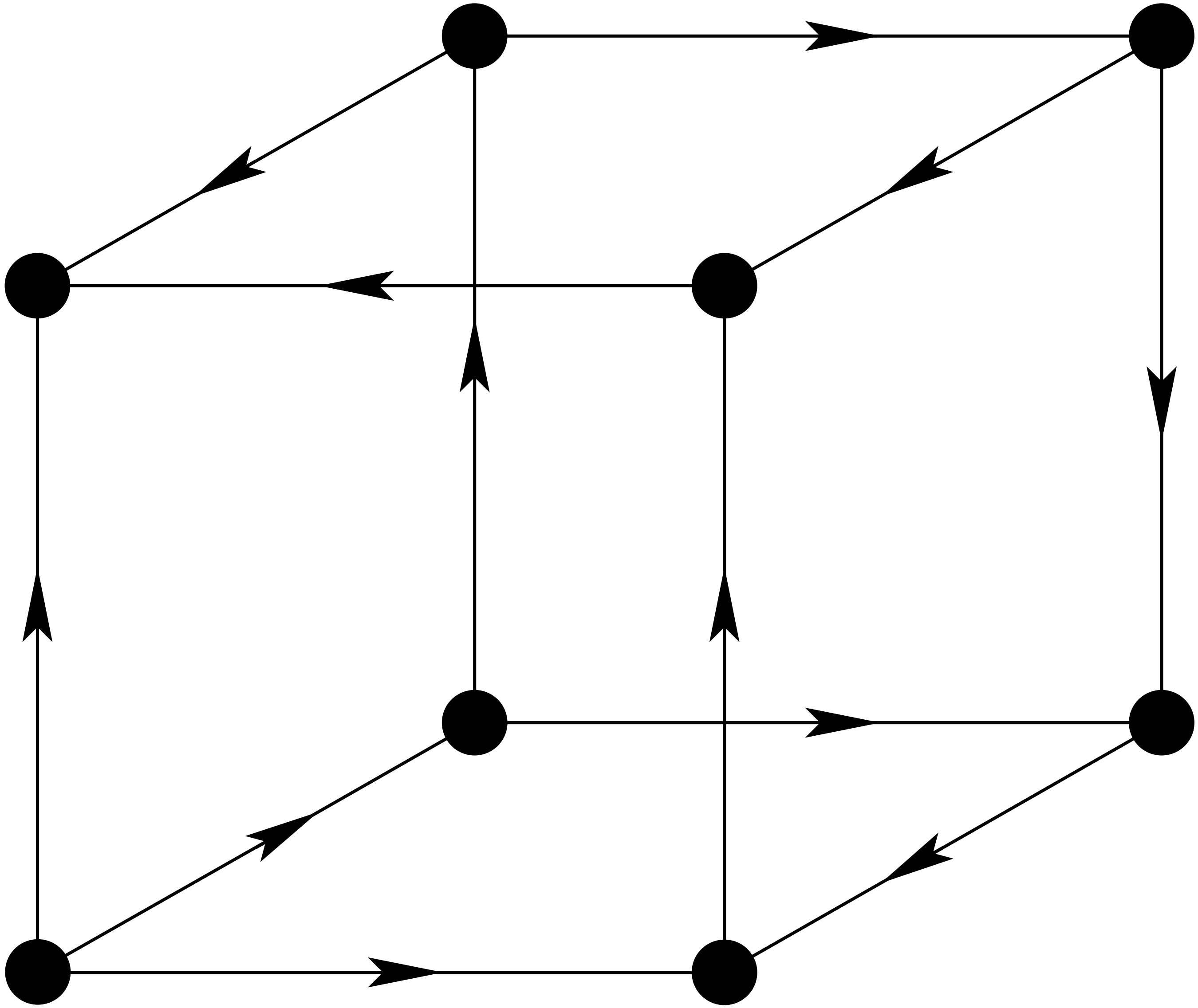
Completely Unimodal Hypercubes

A PBF is <u>completely unimodal</u> (CU) if it has exactly one maximum on every face of the hypercube.

Completely unimodal functions are also known as

- Completely Unimodal numberings, and
- Acyclic Unique Sink Orders.

CU Hypercubes – Example



CU Hypercubes – Properties

- All local optima are global
- A sufficient condition is for all 2-faces to be Completely Unimodal
- A CU numbering corresponds to a shelling of the dual polytope
- An n-dimensional CU Hypercube satisfies the Hirsch Conjecture. That is, from every vertex there is a path of length $\leq n$ to the global maximum.
- The Vector of Improving Directions is injective.

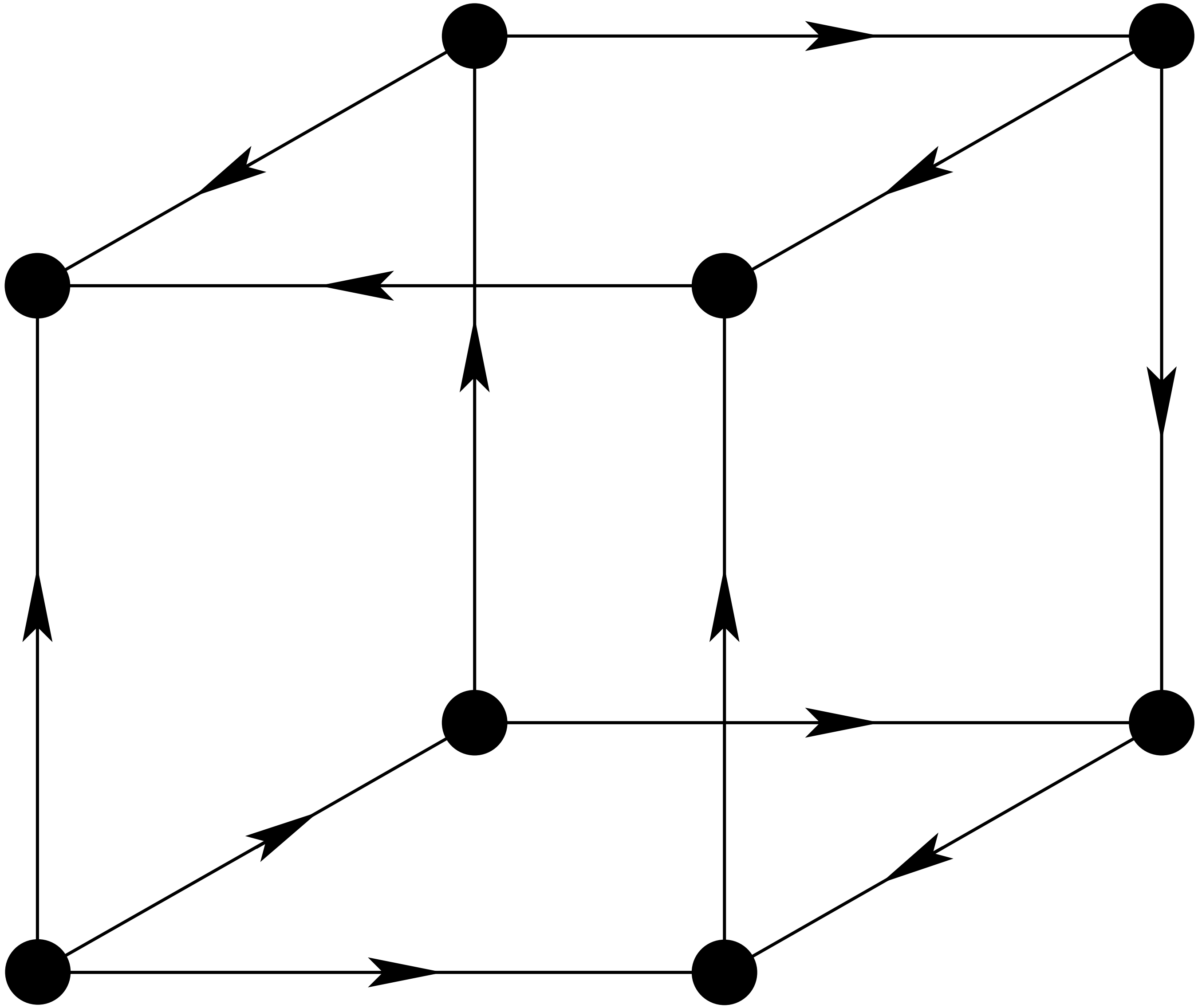
CU Hypercubes – Algorithms

Algorithms to find the global maximum:

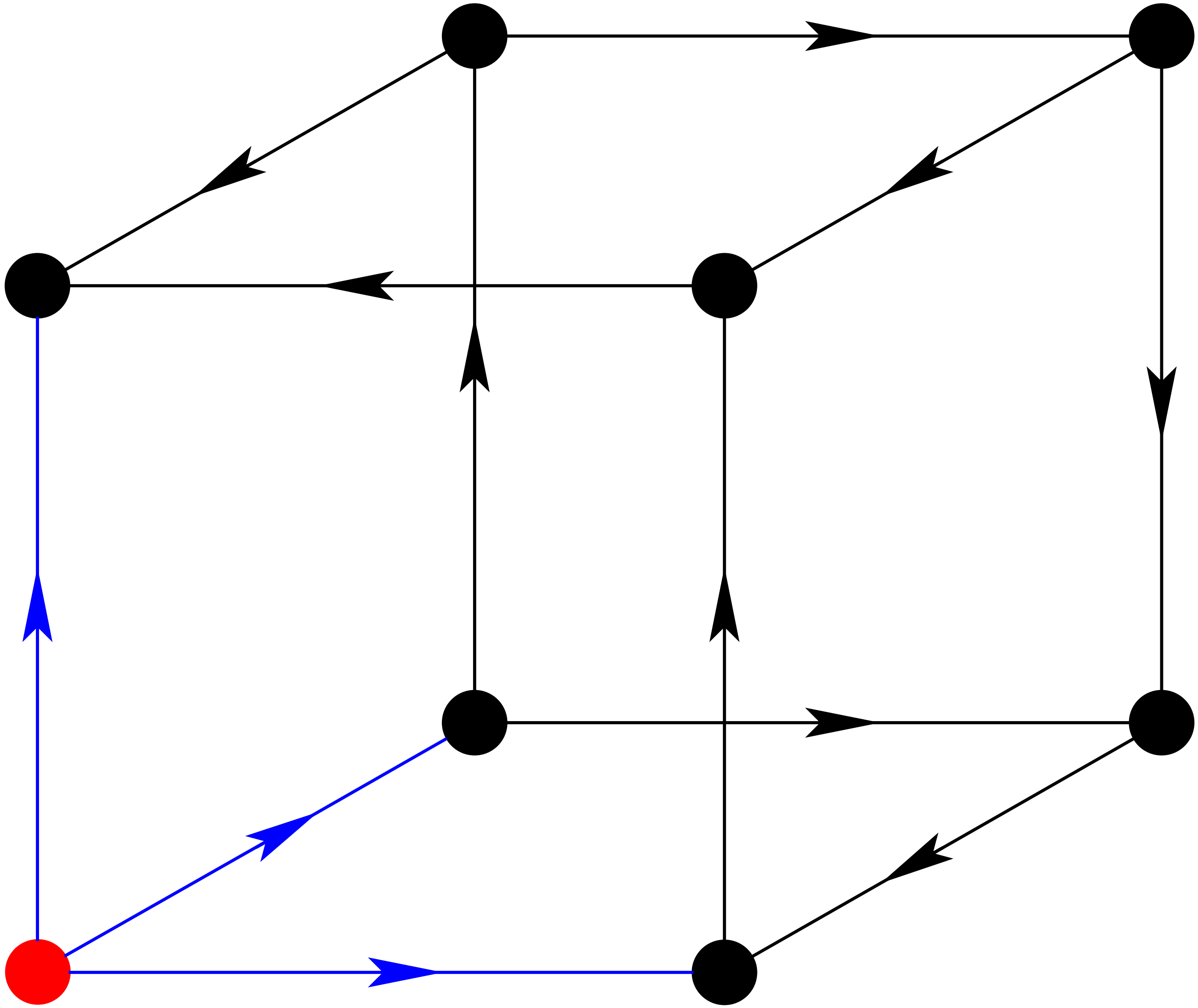
Greedy Local Improvement (GLI): While there are better neighbours of the current position, change in all co-ordinates that are improving.
The complete unimodality condition guarantees that every change results in an improved position.

Fibonacci See-Saw (FSS): Store the maxima of opposite i-faces as i goes from 0 to n. To proceed from i to $i+1$ choose a direction which is improving for only one maximum (such a direction exists by the injectivity of the VID).

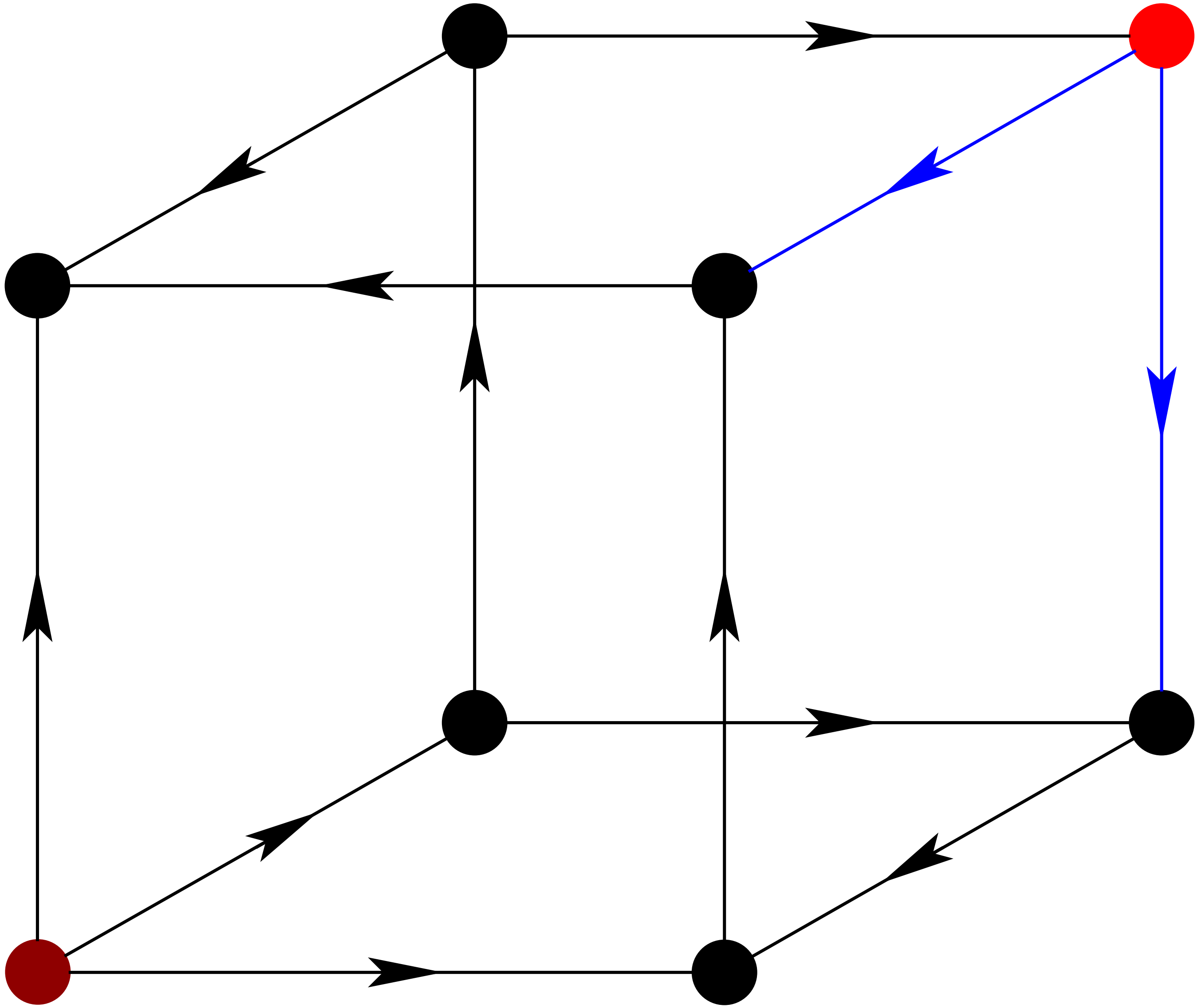
CU Hypercubes – GLI Example



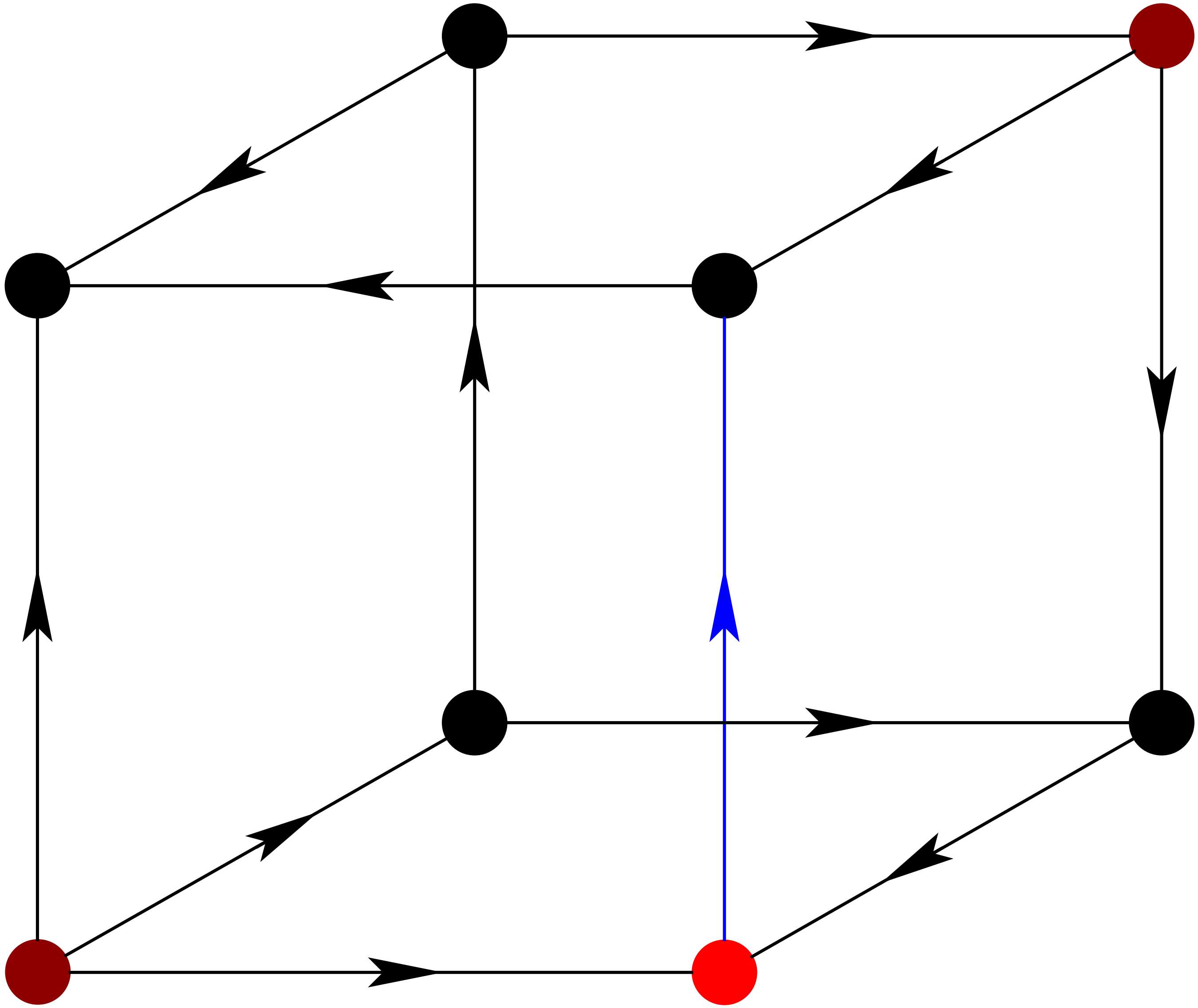
CU Hypercubes – GLI Example



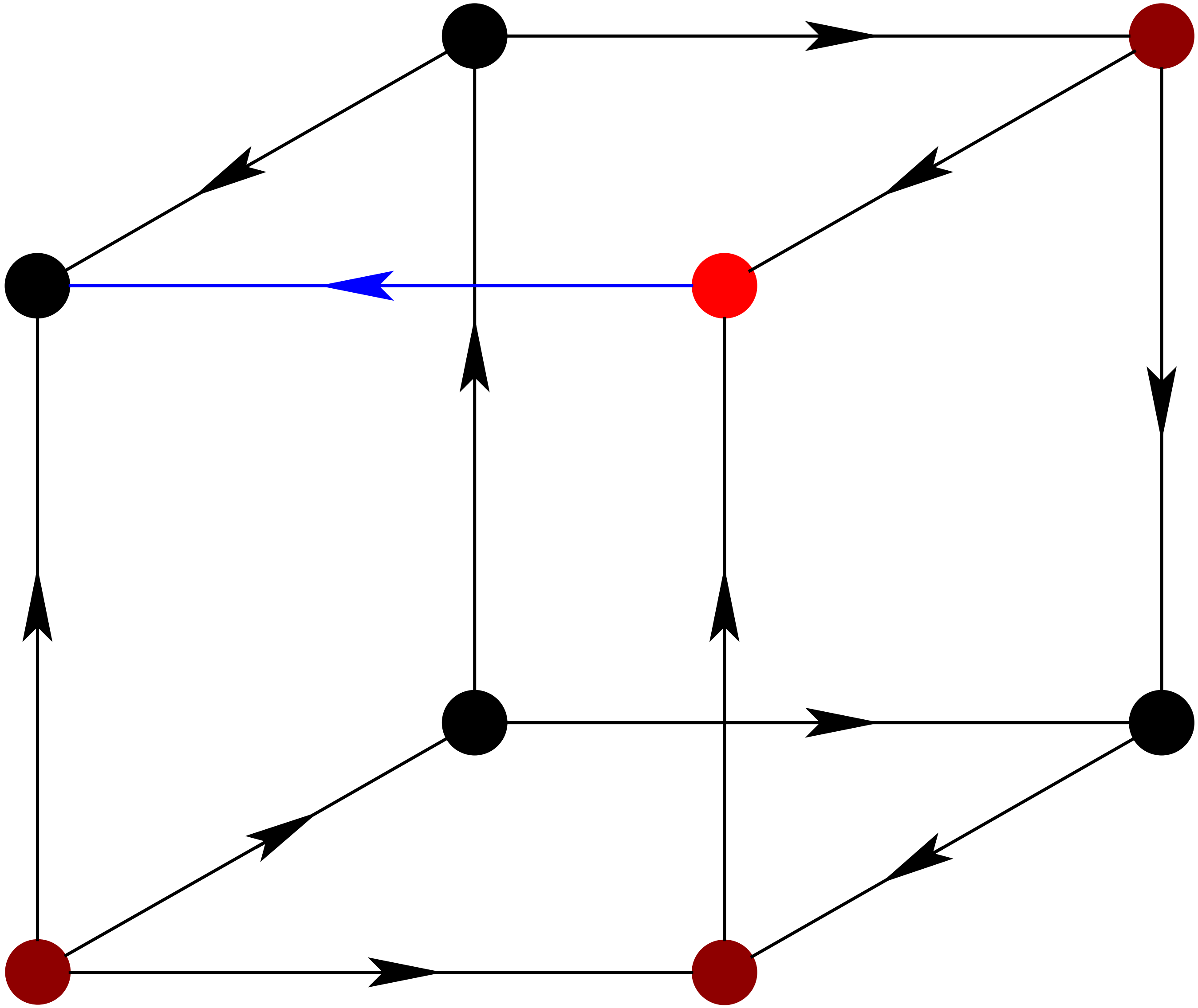
CU Hypercubes – GLI Example



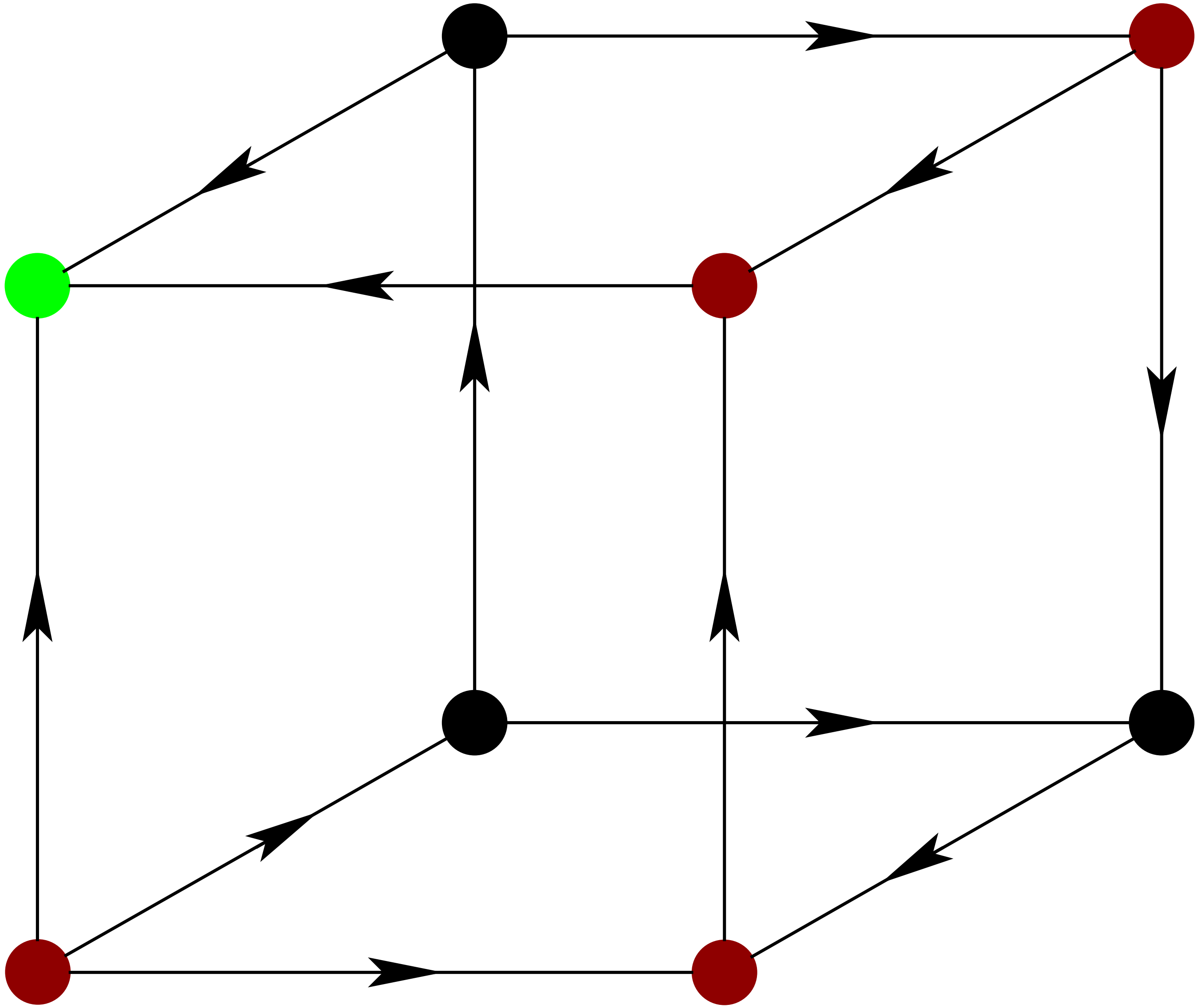
CU Hypercubes – GLI Example



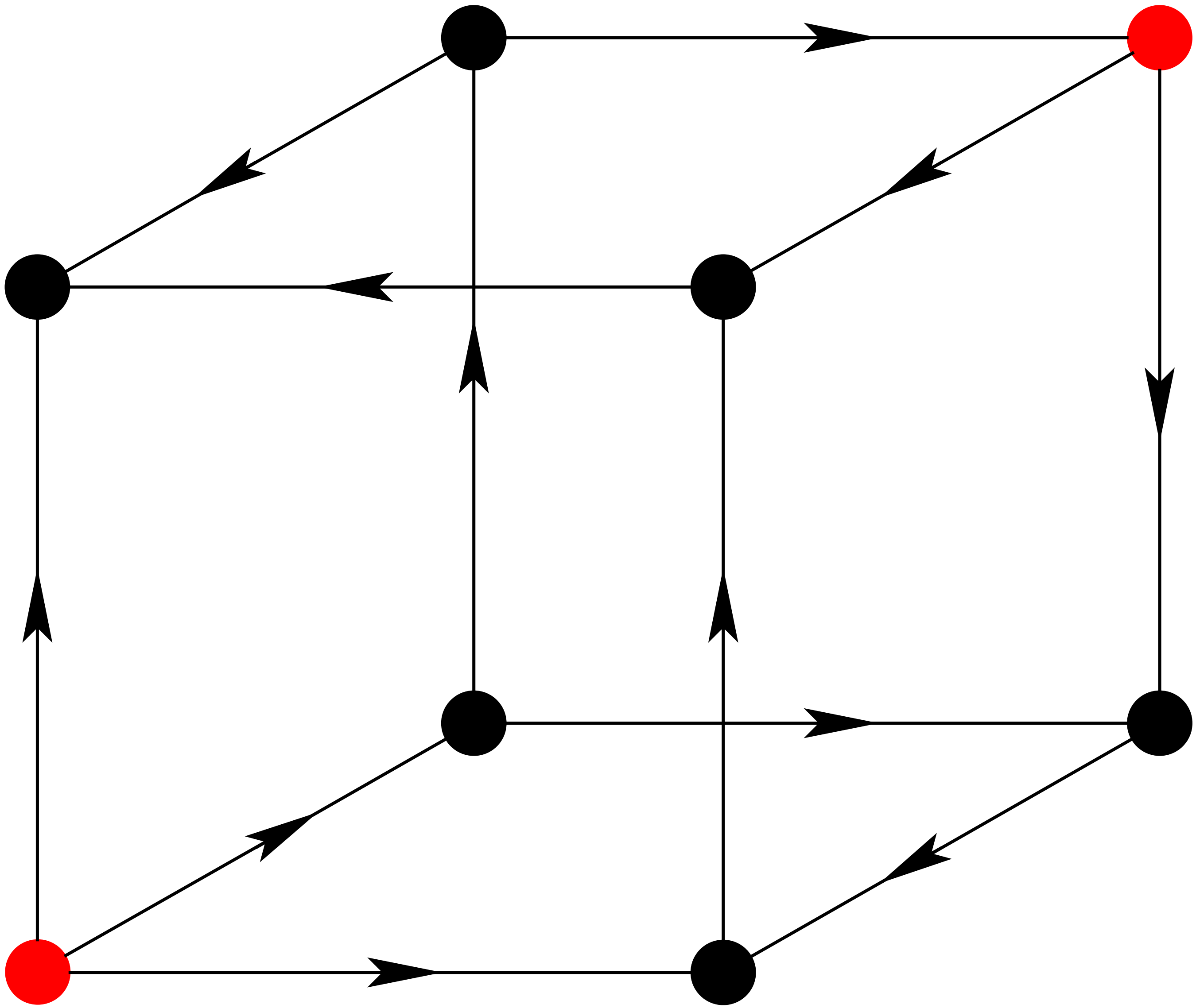
CU Hypercubes – GLI Example



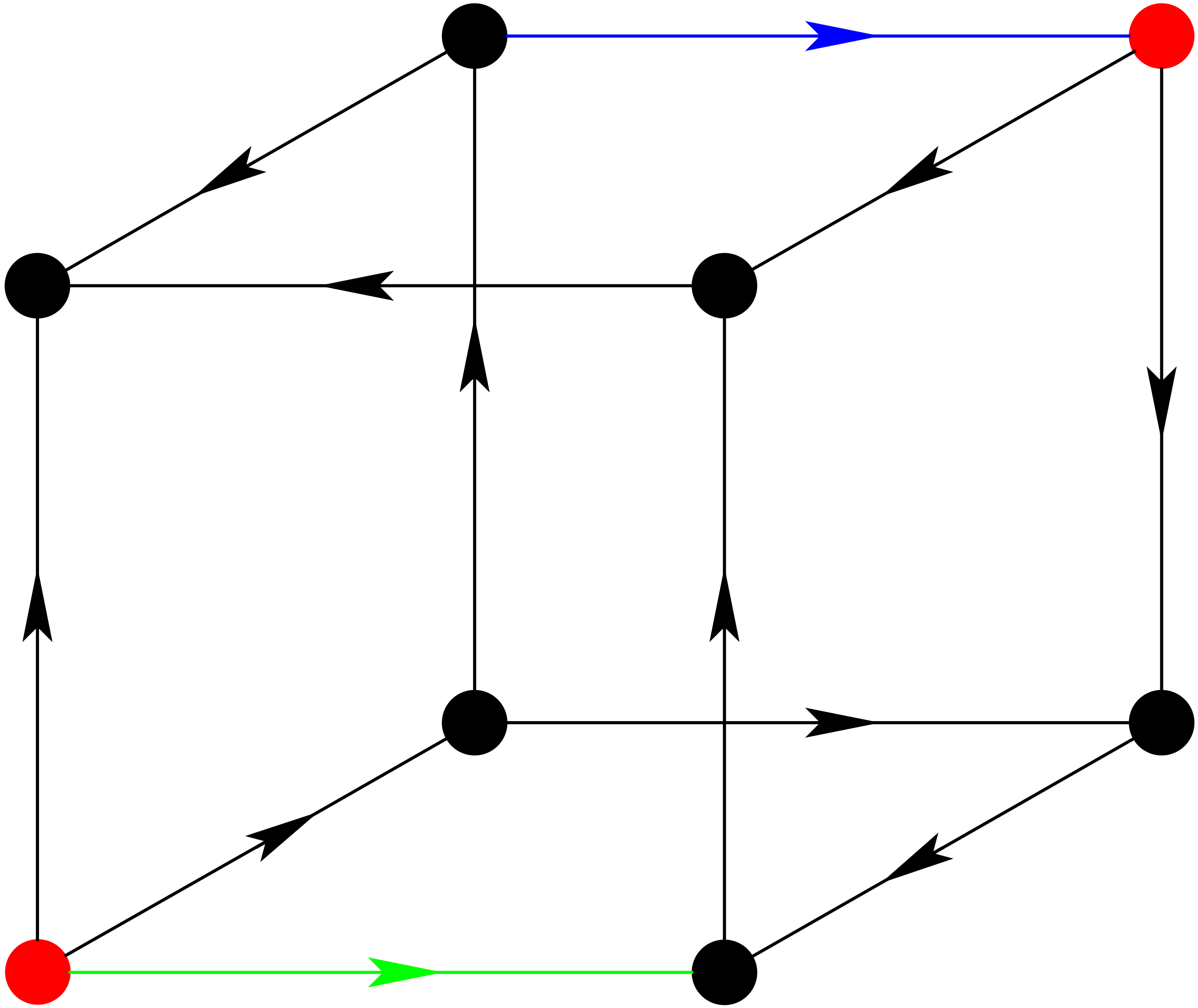
CU Hypercubes – GLI Example



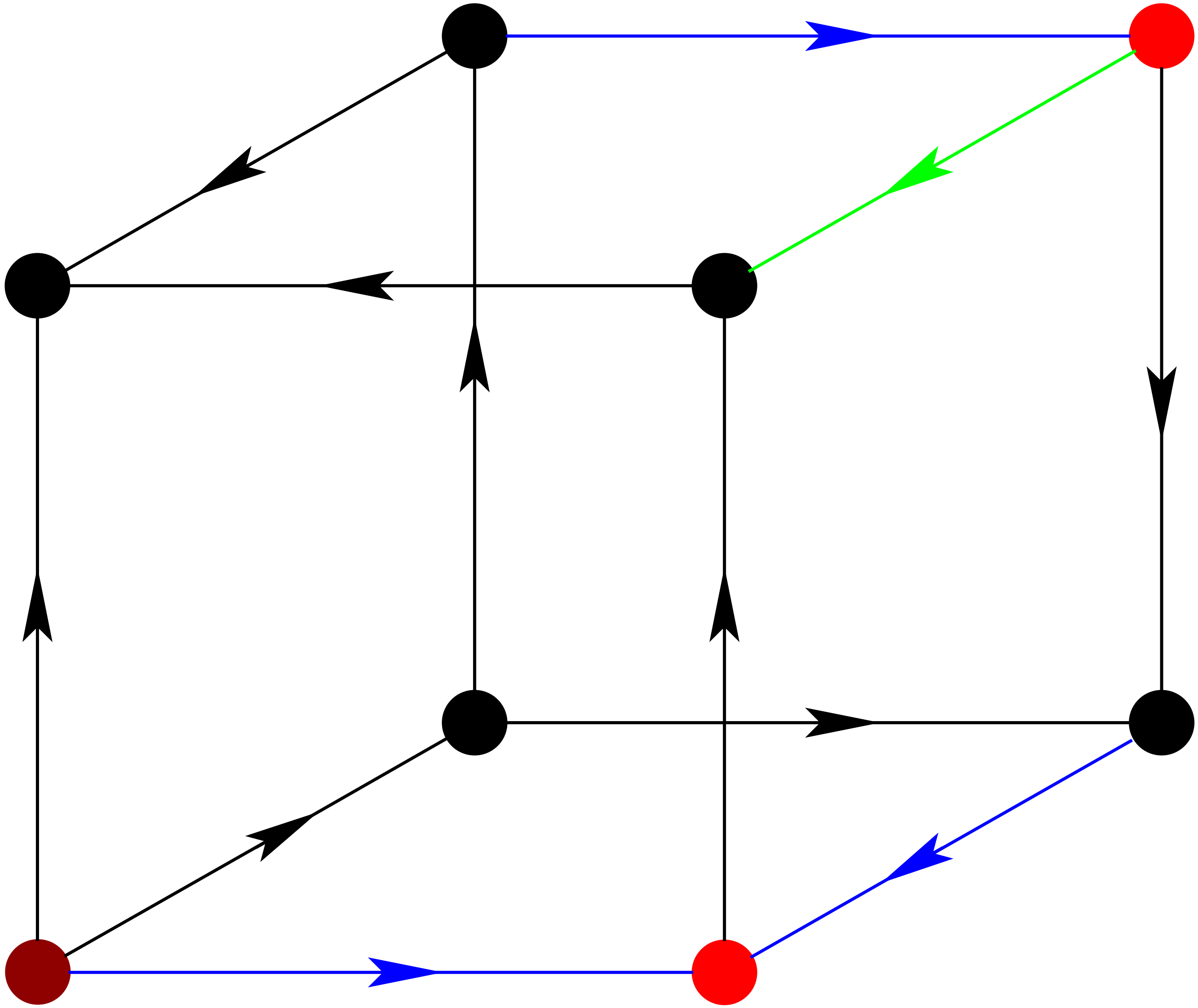
CU Hypercubes – FSS Example



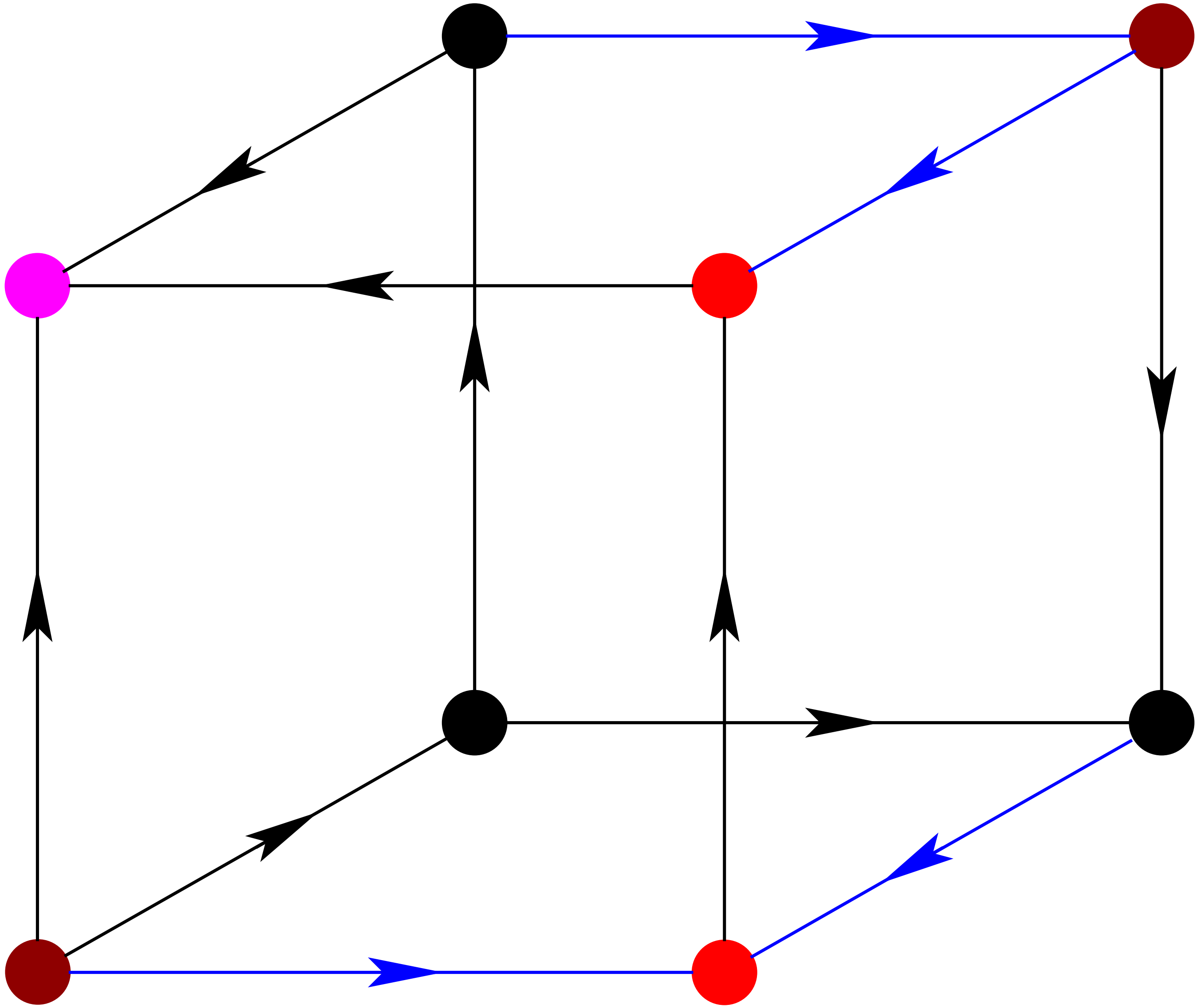
CU Hypercubes – FSS Example



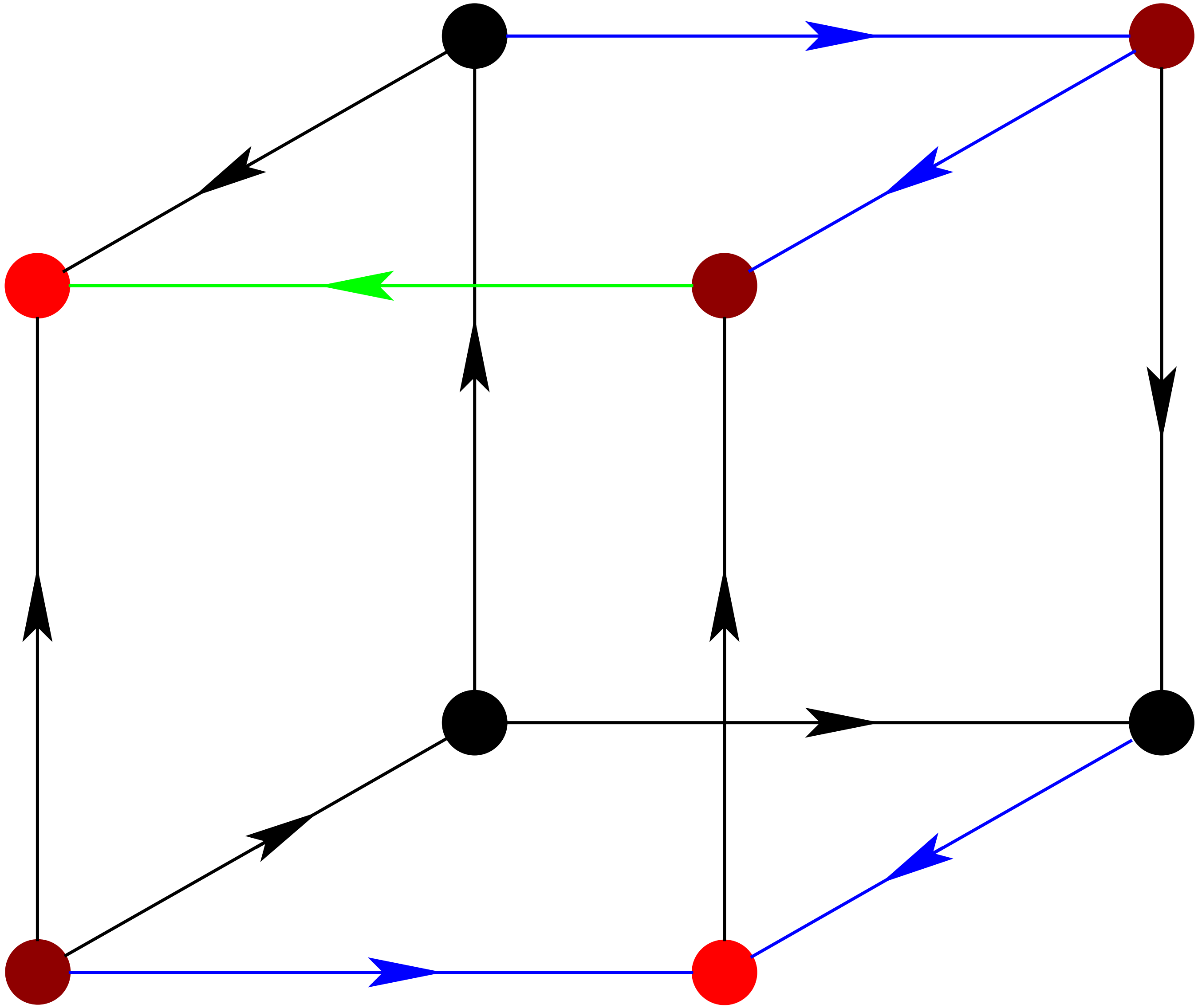
CU Hypercubes – FSS Example



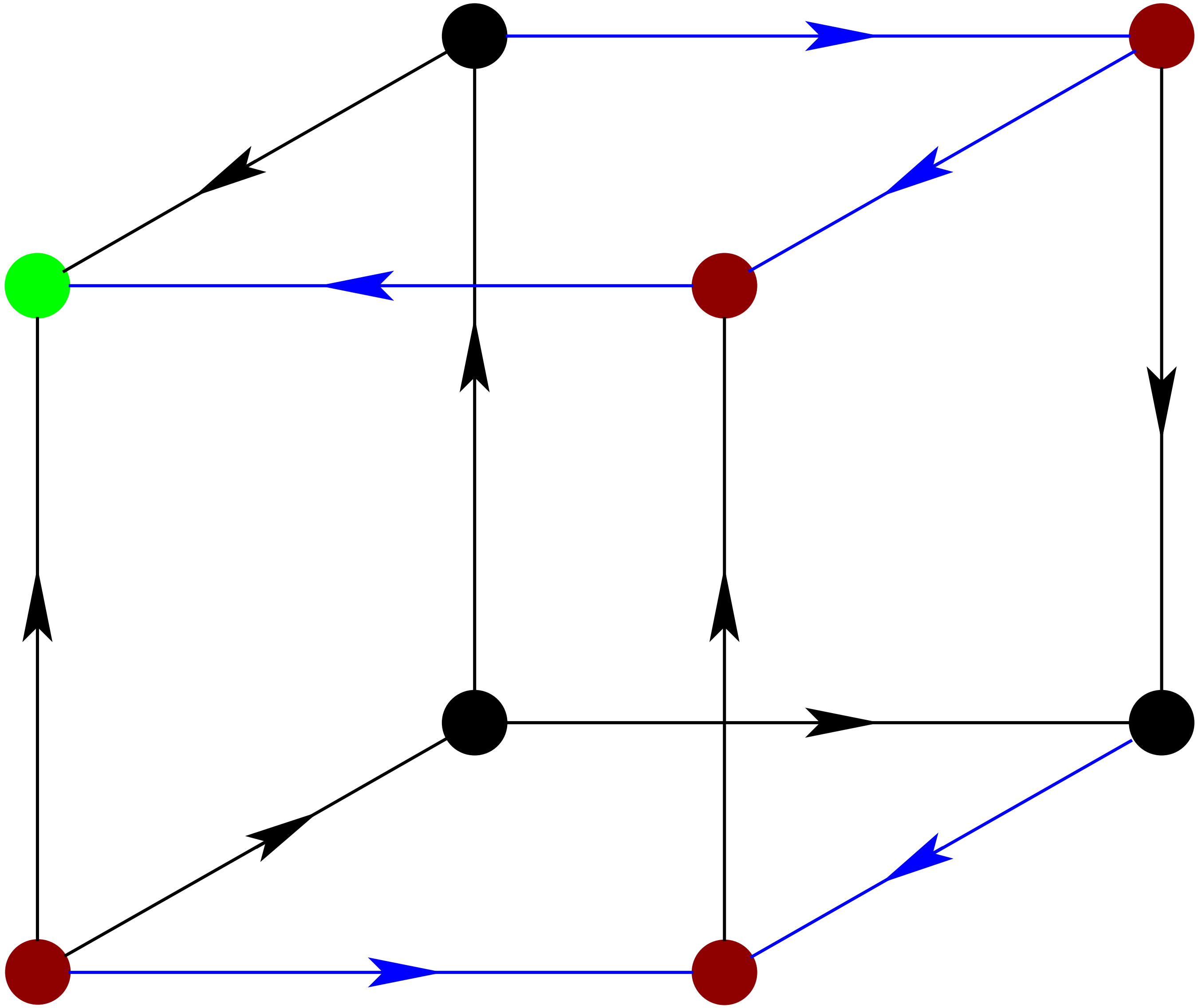
CU Hypercubes – FSS Example



CU Hypercubes – FSS Example



CU Hypercubes – FSS Example



CU Hypercubes – Parity Games

The strategy space of Player 0's strategies forms a hypercube.

Björklund, Sandberg and Vorobyov (2004) showed that the valuation of Vöge and Jurdziński is a CU function on this hypercube.

Their algorithm is then an instance of a GLI.

CU Hypercubes – Problems

Question: *What are the bounds for a GLI?*

Question: *Does every GLI arise from an instance of the Strategy Improvement algorithm?*

Results

Results

Upper bounds: Find necessary conditions for GLI

Lower bounds: Find sufficient conditions for GLI

Results

Upper bounds: Find necessary conditions for GLI

Lower bounds: Find sufficient conditions for GLI

Notation: If $x_0, x_1, \ldots$ is a sequence of hypercube vertices,

- Δ_{ij} is the set of co-ordinates on which x_i and x_j differ
- $\Delta_i := \Delta_{i(i+1)}$

Results – Necessary conditions

Mansour and Singh (1999):

- $\Delta_i \not\subseteq \Delta_j$ for $i < j$
- There are at least $|\Delta_i|$ hypercube vertices valued between x_i and x_{i+1}

These imply that a GLI has at most $O(\frac{2^n}{n})$ steps.

Results – Necessary conditions

Mansour and Singh (1999):

- $\Delta_i \not\subseteq \Delta_j$ for $i < j$
- There are at least $|\Delta_i|$ hypercube vertices valued between x_i and x_{i+1}

These imply that a GLI has at most $O(\frac{2^n}{n})$ steps.

Madani (1999), H. (2004):

- For $i < j$, x_j is not in the face defined at x_i by the directions not improving at x_i ($\Delta_{ij} \not\subseteq \overline{\Delta_i}$)

This implies a GLI has at most 2^{n-1} steps.

Results – Necessary conditions

H. (2004):

PI: For $i < j$, $\Delta_i \cap \Delta_{ij} \not\subseteq \Delta_j$

Implies first condition of Mansour and Singh as well as condition of Madani.

Results – Necessary conditions

H. (2004):

PI: For $i < j$, $\Delta_i \cap \Delta_{ij} \not\subseteq \Delta_j$

Implies first condition of Mansour and Singh as well as condition of Madani.

Question: *What are the bounds for a PI sequence?*

Results – Necessary conditions

H. (2004):

PI: For $i < j$, $\Delta_i \cap \Delta_{ij} \not\subseteq \Delta_j$

Implies first condition of Mansour and Singh as well as condition of Madani.

Question: *What are the bounds for a PI sequence?*

Dimension	1	2	3	4	5	6	7
Longest PI sequence	2	3	5	8	13	21	$\geq$ 26

Conjecture: n*-dimensional PI sequences are bounded by* F_{n+1} *and this bound is attained.*

Results – Sufficient conditions

Conjecture: *PI is sufficient for GLI.*

Results – Other

FSS has worst case running time F_{n+1} (Szabó and Welzl, 2001)

Question: *Is this bound attained?*

Question: *Does this worst case coincide with that of PI?*

Conclusion

Conclusion

- Identified several algorithms (Strategy Improvement, GLI, PI) for which upper and lower bounds remain elusive

Conclusion

- Identified several algorithms (Strategy Improvement, GLI, PI) for which upper and lower bounds remain elusive
- Improved bound on Strategy Improvement algorithm to $O(|E|2^{|V|})$

Conclusion

- Identified several algorithms (Strategy Improvement, GLI, PI) for which upper and lower bounds remain elusive
- Improved bound on Strategy Improvement algorithm to $O(|E|2^{|V|})$
- Can improve Strategy Improvement algorithm to $O(|V||E|F_{|V|}) = O(|V||E|(1.62)^{|V|})$ by using FSS

Conclusion

- Identified several algorithms (Strategy Improvement, GLI, PI) for which upper and lower bounds remain elusive
- Improved bound on Strategy Improvement algorithm to $O(|E|2^{|V|})$
- Can improve Strategy Improvement algorithm to $O(|V||E|F_{|V|}) = O(|V||E|(1.62)^{|V|})$ by using FSS
- Conjectured that the complexity of the Strategy Improvement algorithm is $O(|V||E|F_{|V|})$, and this bound is attained.

One last thing....

One last thing....

One last thing....

One last thing....

One last thing....

One last thing....

One last thing....

One last thing....

One last thing....

One last thing....

One last thing....

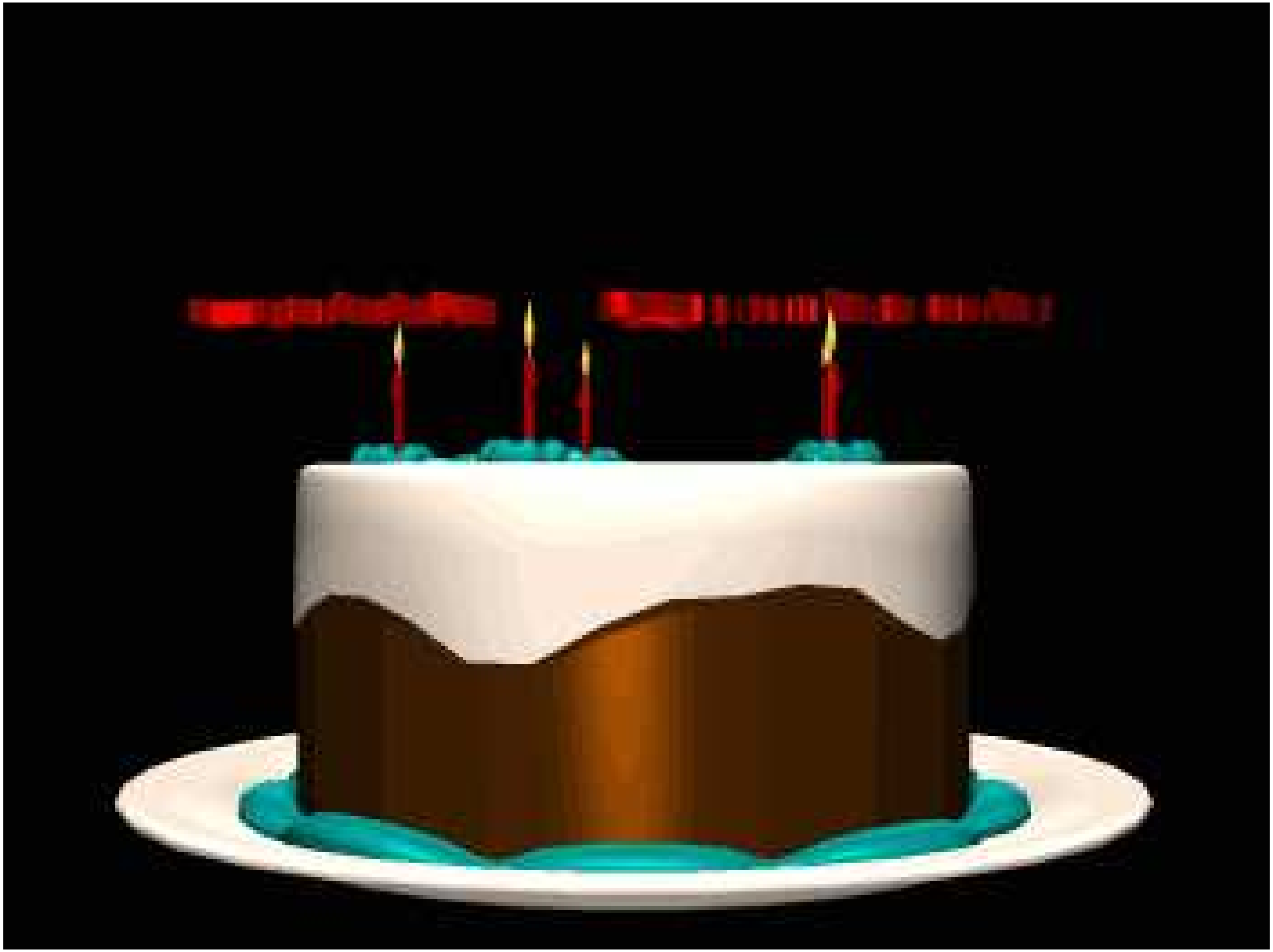
One last thing....

One last thing....

One last thing....

One last thing....

One last thing....



One last thing....

One last thing....

One last thing....

One last thing....

One last thing....