Faster Algorithms for Quantitative Verification in Constant Treewidth Graphs

Krishnendu Chatterjee, Rasmus Ibsen-Jensen, **Andreas Pavlogiannis**

July 22, 2015

Verification of quantitative properties

- System given explicitly as a finite transition system
- Edges labeled with quantities
  - Resource consumption/production
  - Quality/performance measures
  - Rewards/costs
  - ...
  - Elapsed time
  - Rates of events
- Many applications
  - Resource scheduling problems
  - Energy consumption in embedded systems
  - Execution time, average delay
  - ...
  - Competitive analysis
  - Robustness analysis

Verification of quantitative properties

- System given explicitly as a finite transition system
- Edges labeled with quantities
  - Resource consumption/production
  - Quality/performance measures
  - Rewards/costs
  - ...
  - Elapsed time
  - Rates of events
- Many applications
  - Resource scheduling problems
  - Energy consumption in embedded systems
  - Execution time, average delay
  - ...
  - Competitive analysis
  - Robustness analysis

Verification of quantitative properties

- System given explicitly as a finite transition system
- Edges labeled with quantities
  - Resource consumption/production
  - Quality/performance measures
  - Rewards/costs
  - ...
  - Elapsed time
  - Rates of events
- Many applications
  - Resource scheduling problems
  - Energy consumption in embedded systems
  - Execution time, average delay
  - ...
  - Competitive analysis
  - Robustness analysis

This work

- Three classical quantitative properties
  1. Ratio
  2. Mean Payoff
  3. Initial Credit for energy
- Algorithmic improvements mainly for systems of small treewidth
  - Structural property
  - Targeted to verification of control-flow graphs
  - Dataflow analysis, VLSI design, Parity games...
- ... but not only

This work

- Three classical quantitative properties
  1. Ratio
  2. Mean Payoff
  3. Initial Credit for energy
- Algorithmic improvements mainly for systems of small treewidth
  - Structural property
  - Targeted to verification of control-flow graphs
  - Dataflow analysis, VLSI design, Parity games...
- ... but not only

This work

- Three classical quantitative properties
  1. Ratio
  2. Mean Payoff
  3. Initial Credit for energy
- Algorithmic improvements mainly for systems of small treewidth
  - Structural property
  - Targeted to verification of control-flow graphs
  - Dataflow analysis, VLSI design, Parity games...
- ... but not only

Outline

1. Problem statements
2. Overview of technical results
3. Treewidth of graphs
4. Illustration
   - Ratio
   - Initial Credit
5. Experimental results

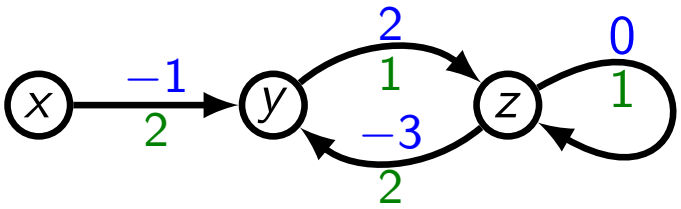
1 Ratio

System given as a graph $G = (V, E)$

- $\mathrm{wt}^1 : E \to \mathbb{Z}$
- $\mathrm{wt}^2 : E \to \mathbb{N}^+$
- Given infinite trace $P : e_1, e_2, \ldots$, the weight of the k-prefix is

$$\mathrm{wt}^i(P^k) = \sum_{j \le k} \mathrm{wt}^i(e_j)$$

- Ratio of trace P: $\overline{\mathrm{wt}}(P) = \lim_{k \to \infty} \inf \frac{\mathrm{wt}^1(P^k)}{\mathrm{wt}^2(P^k)}$
- Ratio property from node u: $\inf_{P_u} \overline{\mathrm{wt}}(P_u)$



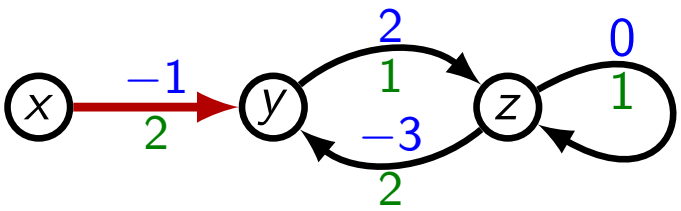
1 Ratio

System given as a graph $G = (V, E)$

- $\mathrm{wt}^1 : E \to \mathbb{Z}$
- $\mathrm{wt}^2 : E \to \mathbb{N}^+$
- Given infinite trace $P : e_1, e_2, \ldots$, the weight of the k-prefix is

$$\mathrm{wt}^i(P^k) = \sum_{j \le k} \mathrm{wt}^i(e_j)$$

- Ratio of trace P: $\overline{\mathrm{wt}}(P) = \lim_{k \to \infty} \inf \frac{\mathrm{wt}^1(P^k)}{\mathrm{wt}^2(P^k)}$
- Ratio property from node u: $\inf_{P_u} \overline{\mathrm{wt}}(P_u)$



$$\frac{-1}{2}$$

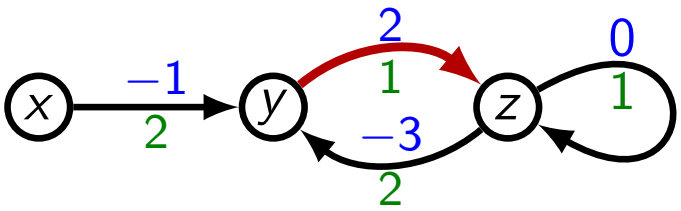
1 Ratio

System given as a graph $G = (V, E)$

- $\mathrm{wt}^1 : E \to \mathbb{Z}$
- $\mathrm{wt}^2 : E \to \mathbb{N}^+$
- Given infinite trace $P : e_1, e_2, \ldots$, the weight of the k-prefix is

$$\mathrm{wt}^i(P^k) = \sum_{j \le k} \mathrm{wt}^i(e_j)$$

- Ratio of trace P: $\overline{\mathrm{wt}}(P) = \lim_{k \to \infty} \inf \frac{\mathrm{wt}^1(P^k)}{\mathrm{wt}^2(P^k)}$
- Ratio property from node u: $\inf_{P_u} \overline{\mathrm{wt}}(P_u)$



$$\frac{-1+2}{2+1}$$

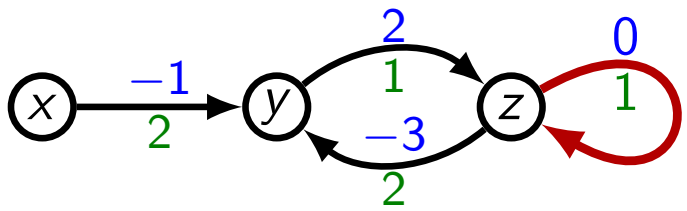
1 Ratio

System given as a graph $G = (V, E)$

- $\mathrm{wt}^1 : E \to \mathbb{Z}$
- $\mathrm{wt}^2 : E \to \mathbb{N}^+$
- Given infinite trace $P : e_1, e_2, \ldots$, the weight of the k-prefix is

$$\mathrm{wt}^i(P^k) = \sum_{j \le k} \mathrm{wt}^i(e_j)$$

- Ratio of trace P: $\overline{\mathrm{wt}}(P) = \lim_{k \to \infty} \inf \frac{\mathrm{wt}^1(P^k)}{\mathrm{wt}^2(P^k)}$
- Ratio property from node u: $\inf_{P_u} \overline{\mathrm{wt}}(P_u)$



$$\frac{-1+2+0}{2+1+1}$$

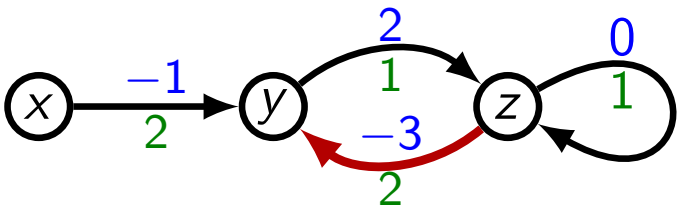
1 Ratio

System given as a graph $G = (V, E)$

- $\mathrm{wt}^1 : E \to \mathbb{Z}$
- $\mathrm{wt}^2 : E \to \mathbb{N}^+$
- Given infinite trace $P : e_1, e_2, \ldots$, the weight of the k-prefix is

$$\mathrm{wt}^i(P^k) = \sum_{j \le k} \mathrm{wt}^i(e_j)$$

- Ratio of trace P: $\overline{\mathrm{wt}}(P) = \lim_{k\to\infty} \inf \frac{\mathrm{wt}^1(P^k)}{\mathrm{wt}^2(P^k)}$
- Ratio property from node u: $\inf_{P_u} \overline{\mathrm{wt}}(P_u)$



$$\frac{-1+2+0-3}{2+1+1+2}$$

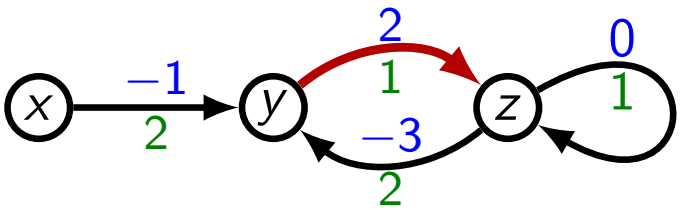
1 Ratio

System given as a graph $G = (V, E)$

- $\mathrm{wt}^1 : E \to \mathbb{Z}$
- $\mathrm{wt}^2 : E \to \mathbb{N}^+$
- Given infinite trace $P : e_1, e_2, \ldots$, the weight of the k-prefix is

$$\mathrm{wt}^i(P^k) = \sum_{j \le k} \mathrm{wt}^i(e_j)$$

- Ratio of trace P: $\overline{\mathrm{wt}}(P) = \lim_{k \to \infty} \inf \frac{\mathrm{wt}^1(P^k)}{\mathrm{wt}^2(P^k)}$
- Ratio property from node u: $\inf_{P_u} \overline{\mathrm{wt}}(P_u)$



$$\frac{-1+2+0-3+2}{2+1+1+2+1}$$

2 Mean Payoff

Special case of Ratio

- $\mathsf{wt}^1 : E \to \mathbb{Z}$
- $\mathsf{wt}^2 : E \to \{1\}$

- Ratio and Mean Payoff witnessed by lasso paths
- For C a simple cycle in G
  - Minimum ratio cycle: $\nu^* = \min_C \frac{\mathsf{wt}^1(C)}{\mathsf{wt}^2(C)}$
  - Minimum mean cycle: $\mu^* = \min_C \frac{\mathsf{wt}^1(C)}{|C|}$

2 Mean Payoff

Special case of Ratio

- $\mathsf{wt}^1 : E \to \mathbb{Z}$
- $\mathsf{wt}^2 : E \to \{1\}$

- Ratio and Mean Payoff witnessed by lasso paths
- For C a simple cycle in G
  - Minimum ratio cycle: $\nu^* = \min_C \frac{\mathsf{wt}^1(C)}{\mathsf{wt}^2(C)}$
  - Minimum mean cycle: $\mu^* = \min_C \frac{\mathsf{wt}^1(C)}{|C|}$

Ratio (constant treewidth)

n nodes $\quad$ W max absolute weight $\quad$ $\nu^* = \frac{a}{b}$

	Burns	Lawler	Our result
Time	$O(n^3)$	$O(n^2 \log(nW))$	$\mathbf{O(n \log \lvert ab \rvert)}$
Space	$O(n)$	$O(n)$	$\mathbf{O(n)}$

Ratio (constant treewidth)

n nodes $\quad$ W max absolute weight $\quad$ $\nu^* = \frac{a}{b}$

	Burns	Lawler	**Our result**
Time	$O(n^3)$	$O(n^2 \log(nW))$	$\mathbf{O(n \log \lvert ab \rvert)}$
Space	$O(n)$	$O(n)$	$\mathbf{O(n)}$

Mean payoff (constant treewidth)

n nodes $\quad$ W max absolute weight $\quad$ $\mu^* = \frac{a}{b}$

	Karp	Orlin & Ahuja	Our result	Our result (ϵ-approx)
Time	$O(n^2)$	$O(n^{1.5} \log(nW))$	$\mathbf{O(n \log \lvert ab \rvert)}$	$\mathbf{O(n \log \frac{n}{\epsilon})}$
Space	$O(n^2)$	$O(n)$	$\mathbf{O(n)}$	$\mathbf{O(n)}$

Mean payoff (constant treewidth)

n nodes $\quad$ W max absolute weight $\quad$ $\mu^* = \frac{a}{b}$

	Karp	Orlin & Ahuja	**Our result**	**Our result (ϵ-approx)**
Time	$O(n^2)$	$O(n^{1.5}\log(nW))$	$\mathbf{O(n \log \lvert ab \rvert)}$	$\mathbf{O(n \log \frac{n}{\epsilon})}$
Space	$O(n^2)$	$O(n)$	$\mathbf{O(n)}$	$\mathbf{O(n)}$

Mean payoff (constant treewidth)

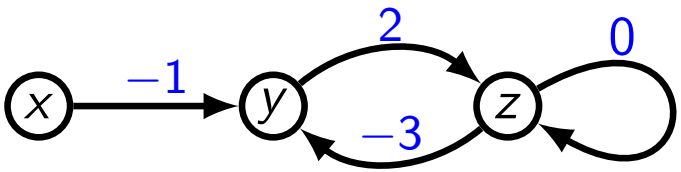
n nodes $\quad$ W max absolute weight $\quad$ $\mu^* = \frac{a}{b}$

	Karp	Orlin & Ahuja	**Our result**	**Our result (ϵ-approx)**
Time	$O(n^2)$	$O(n^{1.5} \log(nW))$	$\mathbf{O(n \log \lvert ab \rvert)}$	$\mathbf{O(n \log \frac{n}{\epsilon})}$
Space	$O(n^2)$	$O(n)$	$\mathbf{O(n)}$	$\mathbf{O(n)}$

3 Initial Credit

- $\mathrm{wt} : E \to \mathbb{Z}$
- Initial credit for energy: give u the smallest initial energy $\mathrm{E}(u) \geq 0$ such that in some infinite trace P_u, the running sum stays non-negative:

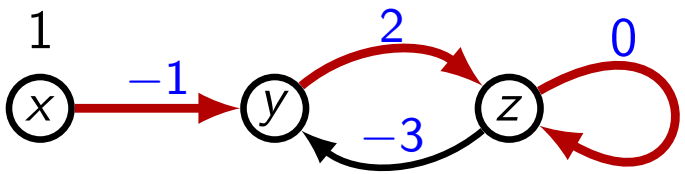
$$\forall k \in \mathbb{N} \quad \mathrm{E}(u) + \mathrm{wt}(P_u^k) \geq 0$$



3 Initial Credit

- $\mathrm{wt} : E \to \mathbb{Z}$
- Initial credit for energy: give u the smallest initial energy $\mathrm{E}(u) \geq 0$ such that in some infinite trace P_u, the running sum stays non-negative:

$$\forall k \in \mathbb{N} \quad \mathrm{E}(u) + \mathrm{wt}(P_u^k) \geq 0$$



Initial credit

n nodes m edges W max absolute weight

	Existing [Bouyer et al.]	Our result (general graphs)	Our result (constant treewidth)
Time (decision)	$O(n^2 m)$	$\mathbf{O(nm)}$	$\mathbf{O(n \log n)}$
Time	$O(n^3 m \log(nW))$	$\mathbf{O(n^2 m)}$	$\mathbf{O(n \log n)}$
Space	$O(m)$	$\mathbf{O(m)}$	$\mathbf{O(m) = O(n)}$

Initial credit

n nodes m edges W max absolute weight

	Existing [Bouyer et al.]	**Our result (general graphs)**	Our result (constant treewidth)
Time (decision)	$O(n^2 m)$	$\mathbf{O(nm)}$	$\mathbf{O(n \log n)}$
Time	$O(n^3 m \log(nW))$	$\mathbf{O(n^2 m)}$	$\mathbf{O(n \log n)}$
Space	$O(m)$	$\mathbf{O(m)}$	$\mathbf{O(m) = O(n)}$

Initial credit

n nodes m edges W max absolute weight

	Existing [Bouyer et al.]	**Our result (general graphs)**	**Our result (constant treewidth)**
Time (decision)	$O(n^2 m)$	$\mathbf{O(nm)}$	$\mathbf{O(n \log n)}$
Time	$O(n^3 m \log(nW))$	$\mathbf{O(n^2 m)}$	$\mathbf{O(n \log n)}$
Space	$O(m)$	$\mathbf{O(m)}$	$\mathbf{O(m) = O(n)}$

Tree Decompositions and Treewidth

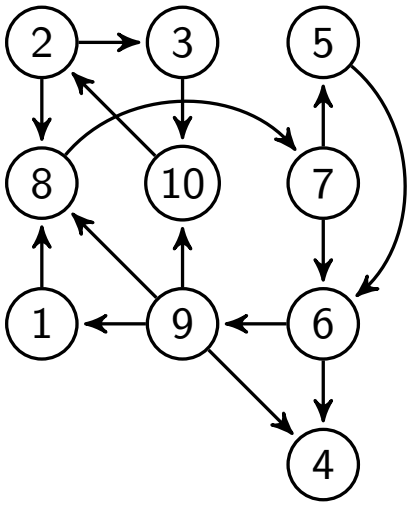
Tree Decomposition

Definition (Tree decomposition)

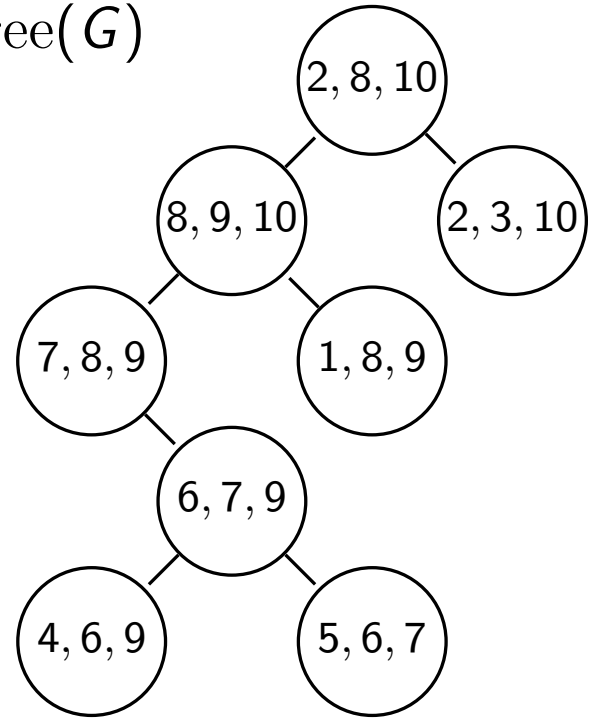
Given a graph $G = (V, E)$, a **tree-decomposition** $\mathrm{Tree}(G) = (V_T, E_T)$ is a tree of bags $B_i \subseteq V$ such that:

1. Every node of G is contained in a bag
2. Every edge of G is contained in a bag
3. Every node of G appears in a contiguous subtree of $\mathrm{Tree}(G)$.

G



$\mathrm{Tree}(G)$



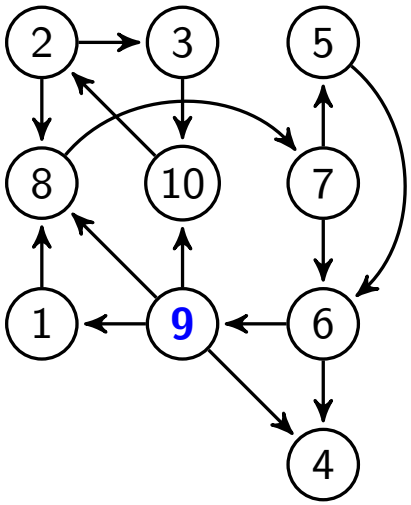
Tree Decomposition

Definition (Tree decomposition)

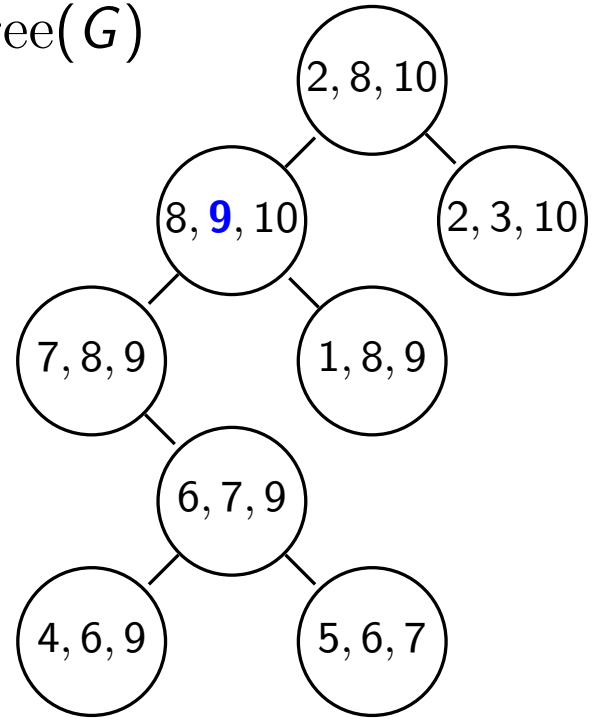
Given a graph $G = (V, E)$, a **tree-decomposition** $\mathrm{Tree}(G) = (V_T, E_T)$ is a tree of bags $B_i \subseteq V$ such that:

1. Every node of G is contained in a bag
2. Every edge of G is contained in a bag
3. Every node of G appears in a contiguous subtree of $\mathrm{Tree}(G)$.

G



$\mathrm{Tree}(G)$



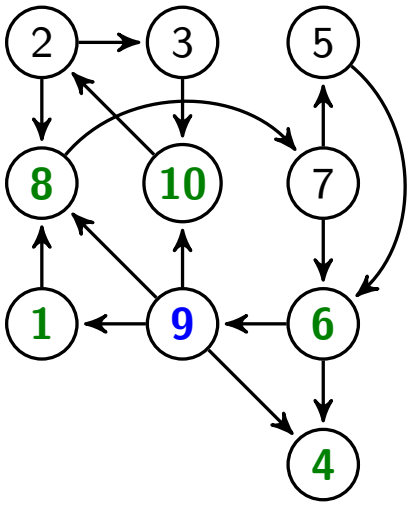
Tree Decomposition

Definition (Tree decomposition)

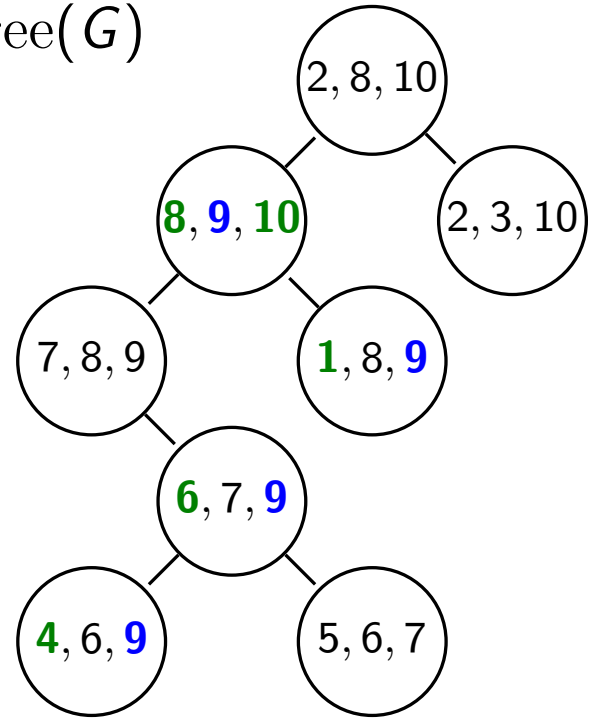
Given a graph $G = (V, E)$, a **tree-decomposition** $\mathrm{Tree}(G) = (V_T, E_T)$ is a tree of bags $B_i \subseteq V$ such that:

1. Every node of G is contained in a bag
2. Every edge of G is contained in a bag
3. Every node of G appears in a contiguous subtree of $\mathrm{Tree}(G)$.

G



$\mathrm{Tree}(G)$



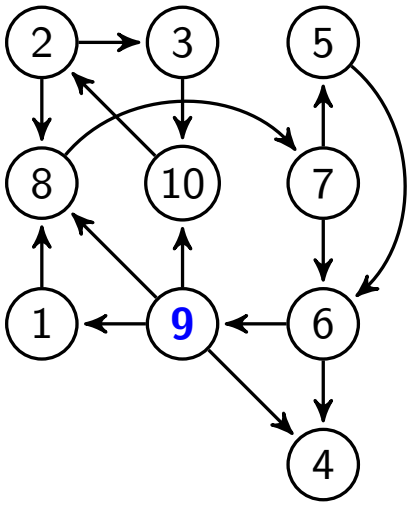
Tree Decomposition

Definition (Tree decomposition)

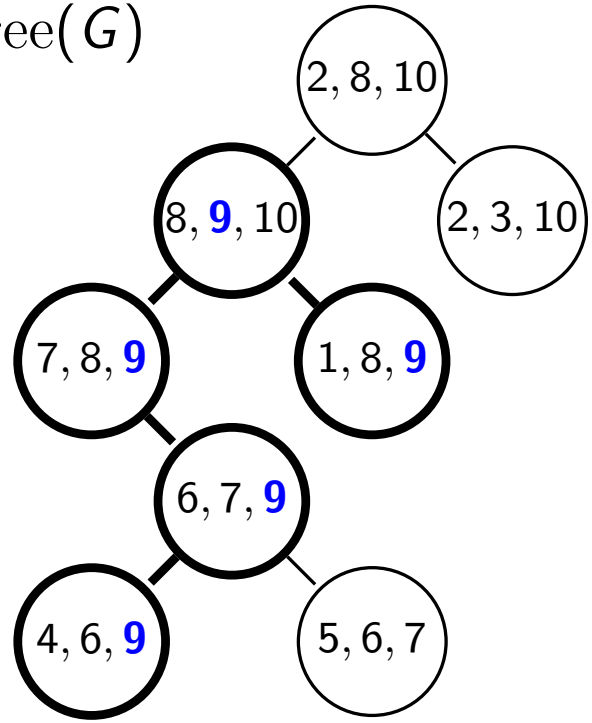
Given a graph $G = (V, E)$, a **tree-decomposition** $\mathrm{Tree}(G) = (V_T, E_T)$ is a tree of bags $B_i \subseteq V$ such that:

1. Every node of G is contained in a bag
2. Every edge of G is contained in a bag
3. Every node of G appears in a contiguous subtree of $\mathrm{Tree}(G)$.

G



$\mathrm{Tree}(G)$



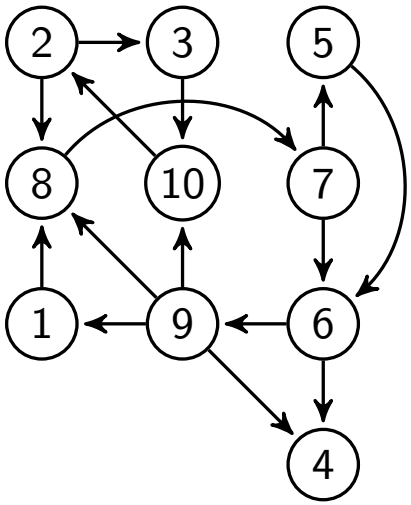
Tree Decomposition

Definition (Tree decomposition)

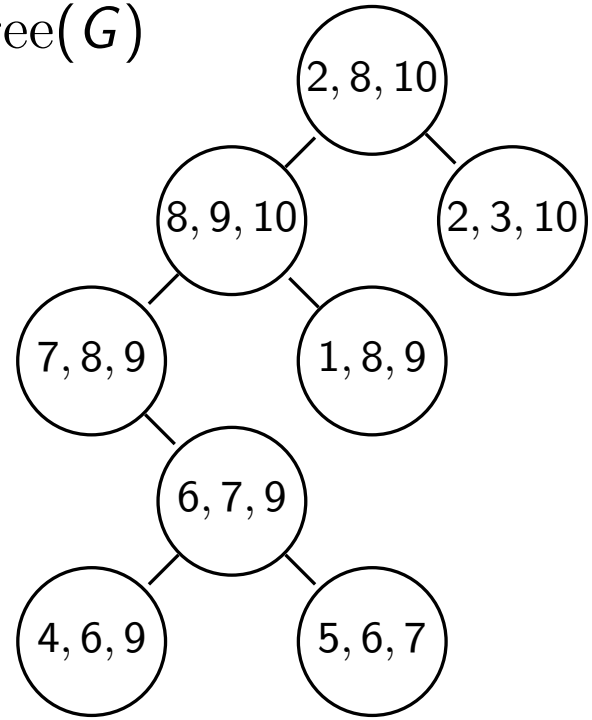
Given a graph $G = (V, E)$, a **tree-decomposition** $\mathrm{Tree}(G) = (V_T, E_T)$ is a tree of bags $B_i \subseteq V$ such that:

1. Every node of G is contained in a bag
2. Every edge of G is contained in a bag
3. Every node of G appears in a contiguous subtree of $\mathrm{Tree}(G)$.

G



$\mathrm{Tree}(G)$



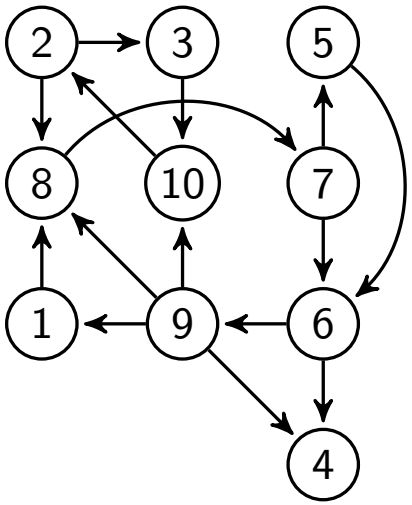
Tree Decomposition

Definition (Tree decomposition)

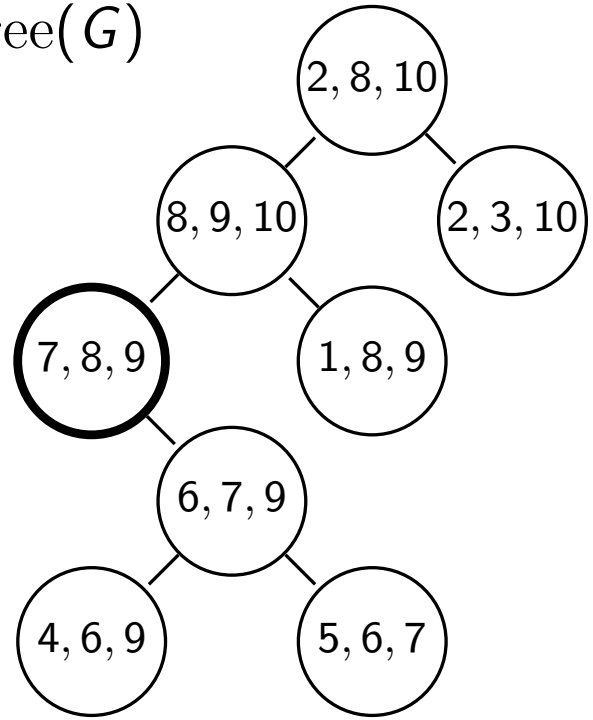
Given a graph $G = (V, E)$, a **tree-decomposition** $\mathrm{Tree}(G) = (V_T, E_T)$ is a tree of bags $B_i \subseteq V$ such that:

1. Every node of G is contained in a bag
2. Every edge of G is contained in a bag
3. Every node of G appears in a contiguous subtree of $\mathrm{Tree}(G)$.

G



$\mathrm{Tree}(G)$



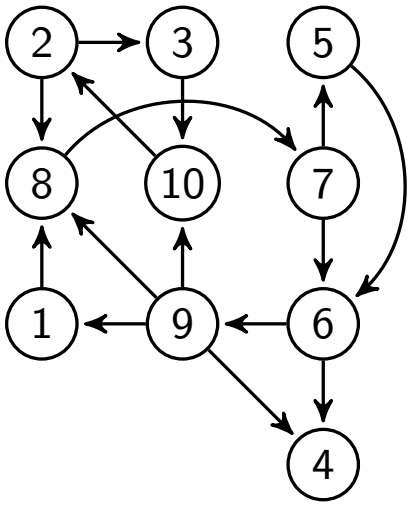
Tree Decomposition

Definition (Tree decomposition)

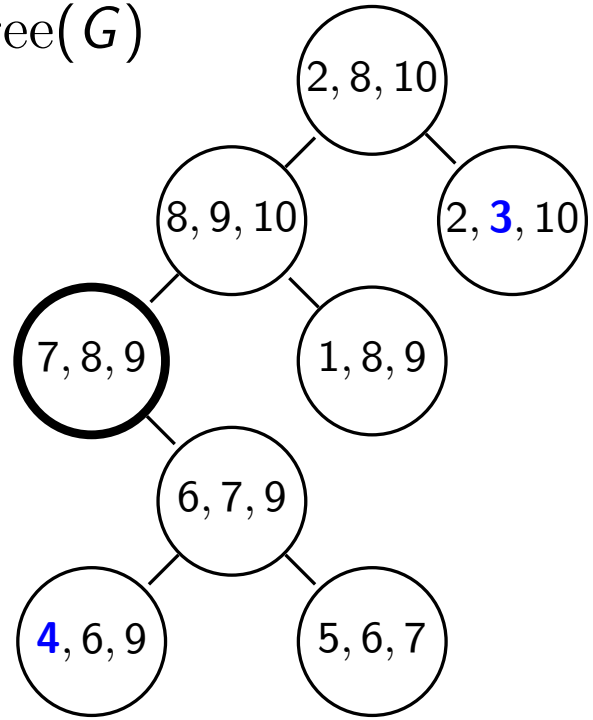
Given a graph $G = (V, E)$, a **tree-decomposition** $\mathrm{Tree}(G) = (V_T, E_T)$ is a tree of bags $B_i \subseteq V$ such that:

1. Every node of G is contained in a bag
2. Every edge of G is contained in a bag
3. Every node of G appears in a contiguous subtree of $\mathrm{Tree}(G)$.

G



$\mathrm{Tree}(G)$



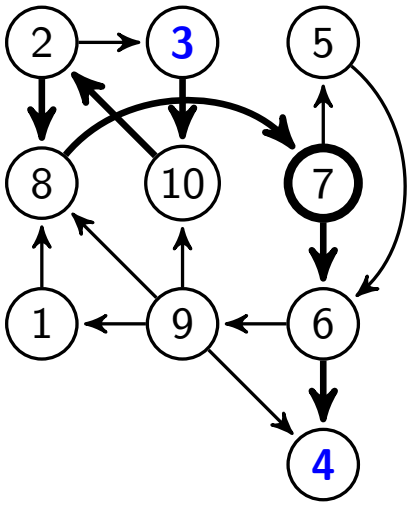
Tree Decomposition

Definition (Tree decomposition)

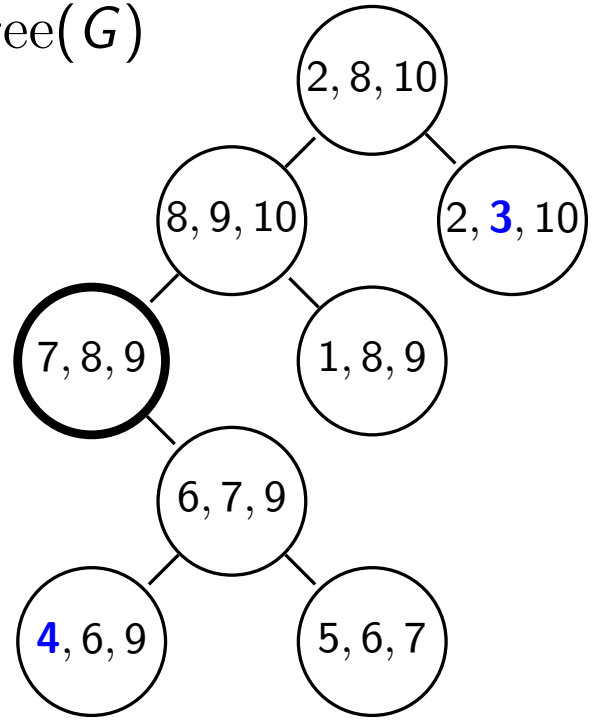
Given a graph $G = (V, E)$, a **tree-decomposition** $\mathrm{Tree}(G) = (V_T, E_T)$ is a tree of bags $B_i \subseteq V$ such that:

1. Every node of G is contained in a bag
2. Every edge of G is contained in a bag
3. Every node of G appears in a contiguous subtree of $\mathrm{Tree}(G)$.

G



$\mathrm{Tree}(G)$

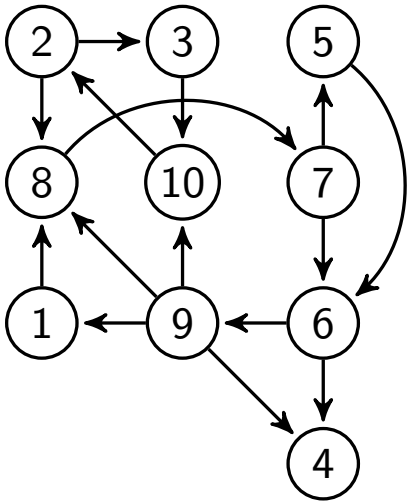


Treewidth

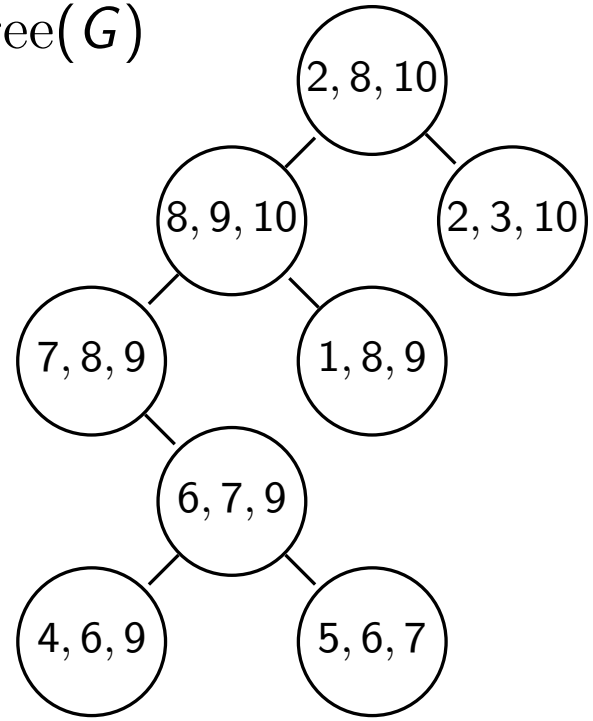
Definition

G has **constant treewidth** if every bag of $\mathrm{Tree}(G)$ has size independent of G.

G



$\mathrm{Tree}(G)$



Treewidth of Structured Programs

Theorem (Tree decomposition)

For constant treewidth graphs, $\mathrm{Tree}(G)$ can be constructed in $O(n)$ time.

Theorem (Treewidth of structured programs)

Control-flow graphs of goto-free programs have constant treewidth.

Ratio

Ratio

$$\nu^* = \min_C \frac{\mathrm{wt}^1(C)}{\mathrm{wt}^2(C)}$$

Ratio

Lemma

Let:

1. ν^* *the ratio value*
2. ν *a guess for* ν^*
3. $\mathsf{wt}_\nu(e) = \mathsf{wt}^1(e) - \nu \cdot \mathsf{wt}^2(e)$

Then $\nu^* \geq \nu$ *iff* $\min_C \mathsf{wt}_\nu(C) \geq 0$ *, where* C *ranges over cycles*

Ratio

Lemma

Let:

1. ν^* *the ratio value*
2. ν *a guess for* ν^*
3. $\mathsf{wt}_\nu(e) = \mathsf{wt}^1(e) - \nu \cdot \mathsf{wt}^2(e)$

Then $\nu^* \geq \nu$ *iff* $\min_C \mathsf{wt}_\nu(C) \geq 0$ *, where* C *ranges over cycles*

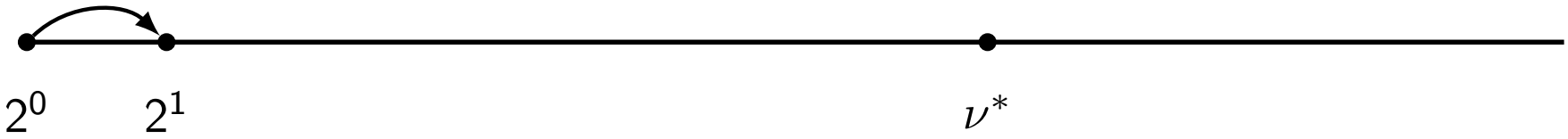
Constant treewidth: $\min_C \mathsf{wt}_\nu(C) \geq 0$ can be detected in $O(n)$ time!

Ratio: Outline

1. Determine $\lfloor \nu^* \rfloor$
   - Exponential search
   - Binary search
     - Stop when $|[\ell, r]| \leq 1$
     - $O(\log \nu^*)$ guesses

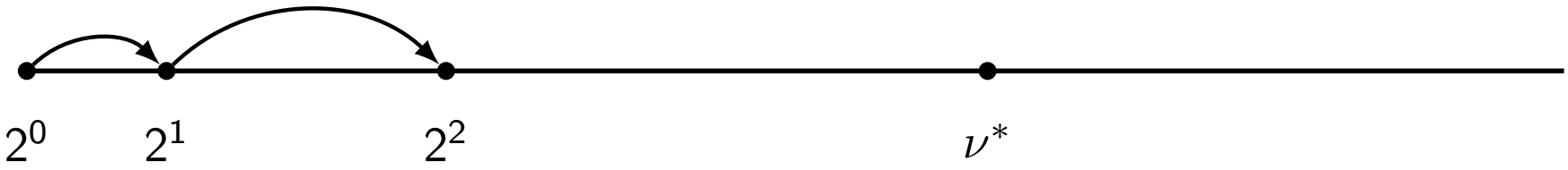
Ratio: Outline

1. Determine $\lfloor \nu^* \rfloor$
   - Exponential search
   - Binary search
     - Stop when $|[\ell, r]| \leq 1$
     - $O(\log \nu^*)$ guesses



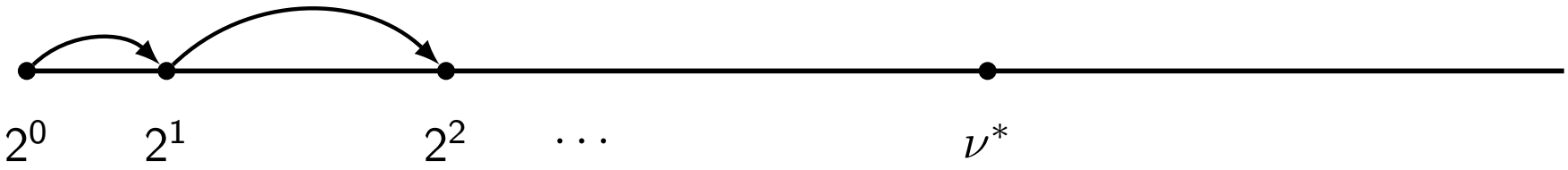
Ratio: Outline

1. Determine $\lfloor \nu^* \rfloor$
   - Exponential search
   - Binary search
     - Stop when $|[\ell, r]| \leq 1$
     - $O(\log \nu^*)$ guesses



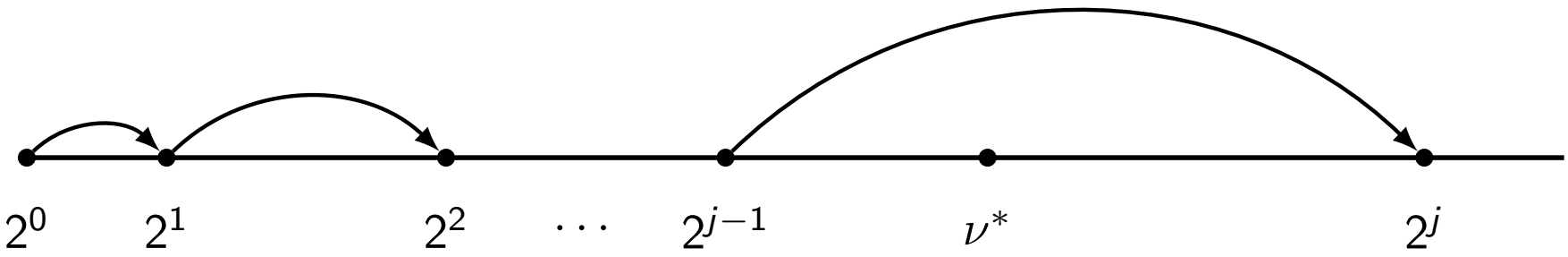
Ratio: Outline

1. Determine $\lfloor \nu^* \rfloor$
   - Exponential search
   - Binary search
     - Stop when $|[\ell, r]| \leq 1$
     - $O(\log \nu^*)$ guesses



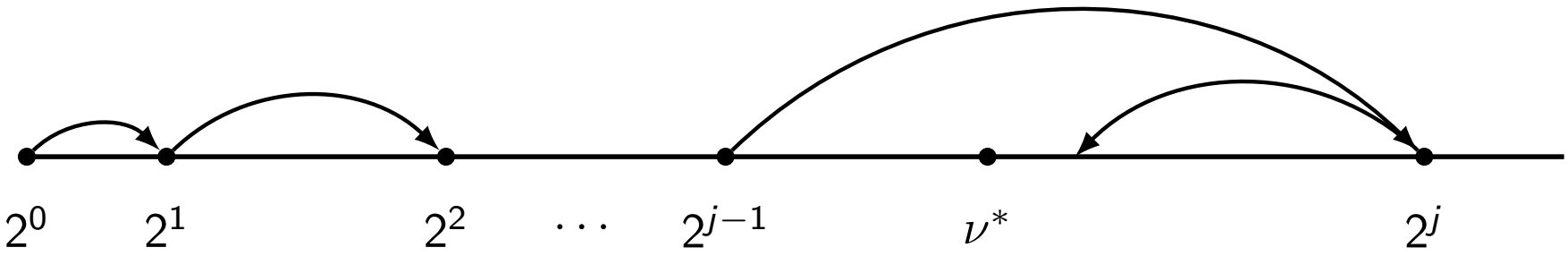
Ratio: Outline

1. Determine $\lfloor \nu^* \rfloor$
   - Exponential search
   - Binary search
     - Stop when $|[\ell, r]| \leq 1$
     - $O(\log \nu^*)$ guesses



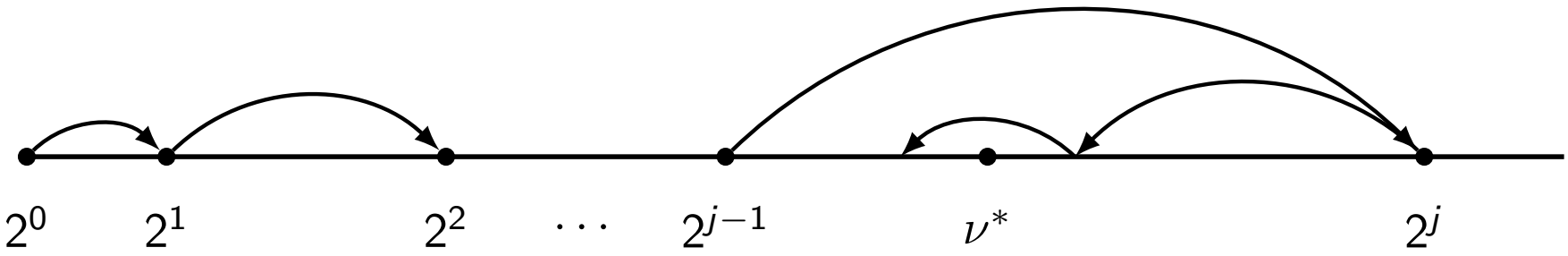
Ratio: Outline

1. Determine $\lfloor \nu^* \rfloor$
   - Exponential search
   - Binary search
     - Stop when $|[\ell, r]| \leq 1$
     - $O(\log \nu^*)$ guesses



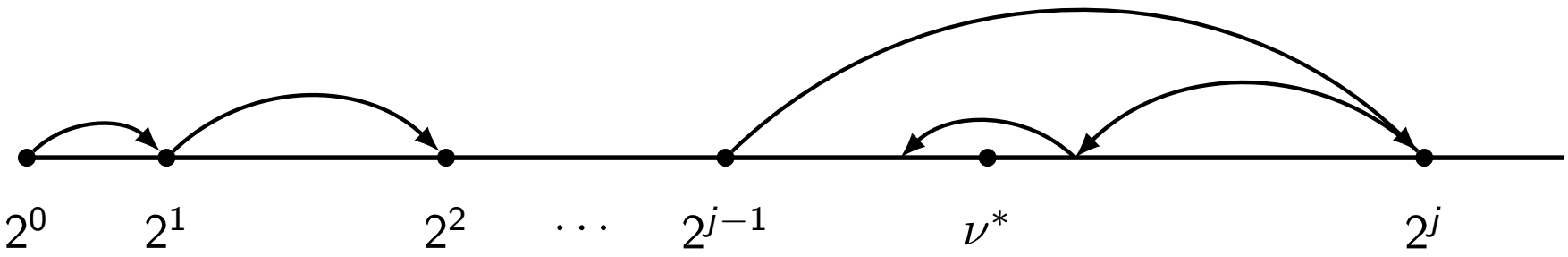
Ratio: Outline

1. Determine $\lfloor \nu^* \rfloor$
   - Exponential search
   - Binary search
     - Stop when $|[\ell, r]| \leq 1$
     - $O(\log \nu^*)$ guesses



Ratio: Outline

1. Determine $\lfloor \nu^* \rfloor$
   - Exponential search
   - Binary search
     - Stop when $|[\ell, r]| \leq 1$
     - $O(\log \nu^*)$ guesses



Ratio: Outline

2. Write $\nu^* = \lfloor \nu^* \rfloor + \frac{x}{b}$
   - Need to determine $\frac{x}{b} < 1$
   - "Efficient search for rationals"
   - Evaluate $O(\log b)$ guesses

Ratio: Outline

2. Write $\nu^* = \lfloor \nu^* \rfloor + \frac{x}{b}$
   - Need to determine $\frac{x}{b} < 1$
   - "Efficient search for rationals"
   - Evaluate $O(\log b)$ guesses

Total guesses: $O(\log \nu^* + \log b)$

Ratio: Outline

2. Write $\nu^* = \lfloor \nu^* \rfloor + \frac{x}{b}$
   - Need to determine $\frac{x}{b} < 1$
   - "Efficient search for rationals"
   - Evaluate $O(\log b)$ guesses

Total guesses: $O(\log \nu^* + \log b) = O(\log(ab))$

$\nu^* = \frac{a}{b}$

Ratio, Mean Payoff (constant treewidth)

n nodes $\quad$ W max absolute weight $\quad$ $\nu^*, \mu^* = \frac{a}{b}$

	Karp	Orlin & Ahuja	**Our result**	**Our result (MP, ϵ-approx)**
Time	$O(n^2)$	$O(n^{1.5} \log(nW))$	$\mathbf{O(n \log \lvert ab \rvert)}$	$\mathbf{O(n \log \frac{n}{\epsilon})}$
Space	$O(n^2)$	$O(n)$	$\mathbf{O(n)}$	$\mathbf{O(n)}$

Initial Credit

Initial credit

Give u the smallest initial energy $\mathsf{E}(u) \geq 0$ such that in some infinite trace P_u, the running sum stays non-negative:

$$\forall k \in \mathbb{N} \quad \mathsf{E}(u) + \mathsf{wt}(P_u^k) \geq 0$$

Initial credit

Give u the largest initial energy $\mathsf{E}(u) \leq 0$ such that in some infinite path P_u, the running sum stays non-positive:

$$\forall k \in \mathbb{N} \quad \mathsf{E}(u) + \mathsf{wt}(P_u^k) \leq 0$$

Initial credit

Give u the largest initial energy $\mathsf{E}(u) \leq 0$ such that in some infinite path P_u, the running sum stays non-positive:

$$\forall k \in \mathbb{N} \quad \mathsf{E}(u) + \mathsf{wt}(P_u^k) \leq 0$$

The distance to a node v:

$$\mathsf{D}_v(u) = \inf_{P : u \rightsquigarrow v} \mathsf{wt}(P)$$

Initial credit

Give u the largest initial energy $\mathsf{E}(u) \leq 0$ such that in some infinite path P_u, the running sum stays non-positive:

$$\forall k \in \mathbb{N} \quad \mathsf{E}(u) + \mathsf{wt}(P_u^k) \leq 0$$

The distance to a node v:

$$\mathsf{D}_v(u) = \inf_{P : u \rightsquigarrow v} \mathsf{wt}(P)$$

The energy to a node v:

$$\mathsf{E}_v(u) = \min_{P : u \rightsquigarrow v} \max_k \mathsf{wt}(P^k)$$

Initial credit

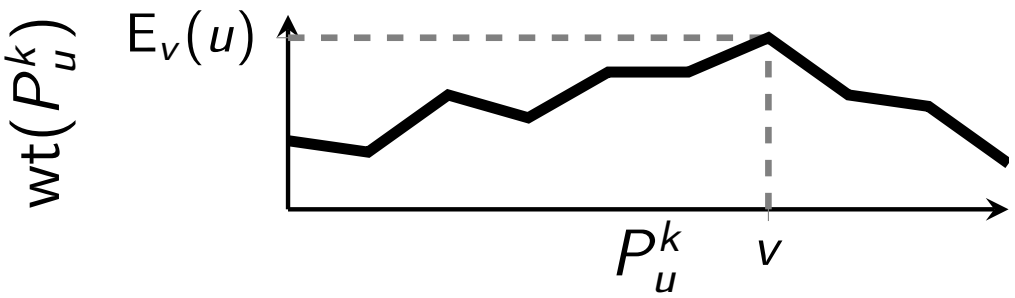
Lemma

For every $u \in V$, we have $\mathsf{E}(u) = \max_{v:\mathsf{E}(v)=0} \mathsf{E}_v(u)$.

Initial credit

Lemma

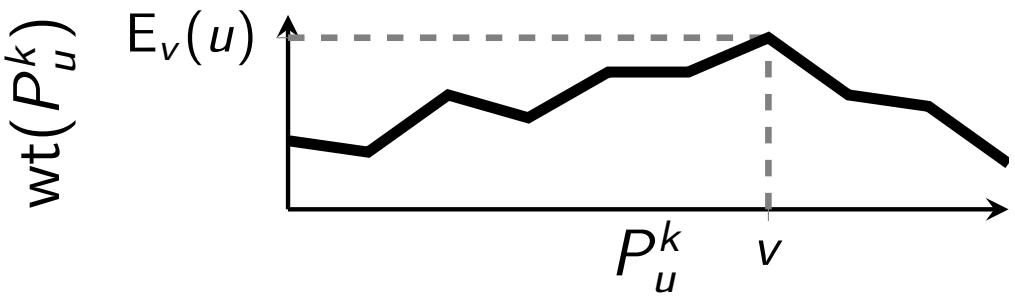
For every $u \in V$, we have $\mathrm{E}(u) = \max_{v:\mathrm{E}(v)=0} \mathrm{E}_v(u)$.



Initial credit

Lemma

For every $u \in V$, we have $\mathrm{E}(u) = \max_{v:\mathrm{E}(v)=0} \mathrm{E}_v(u)$.



Lemma

If $\mathrm{E}_v(u) < 0$ for all u, then $\mathrm{E}_v(u) = -\mathrm{D}_v(u)$ for all u.

The witness path for $\mathrm{D}_v(u)$ attains its maximum at the end

Initial Credit: Outline

Initial Credit: Outline

1. Determine $X = \{v : \mathrm{E}(v) = 0\}$
   - Canceling non-positive cycles

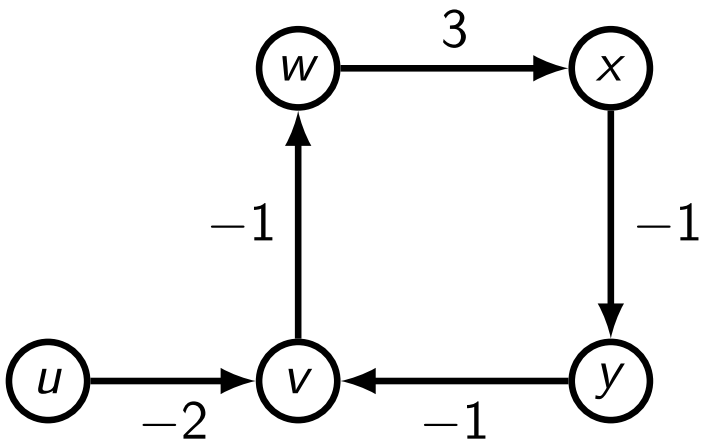
Initial Credit: Outline

1. Determine $X = \{v : \mathsf{E}(v) = 0\}$
   - Canceling non-positive cycles
2. Contract X to a single node z

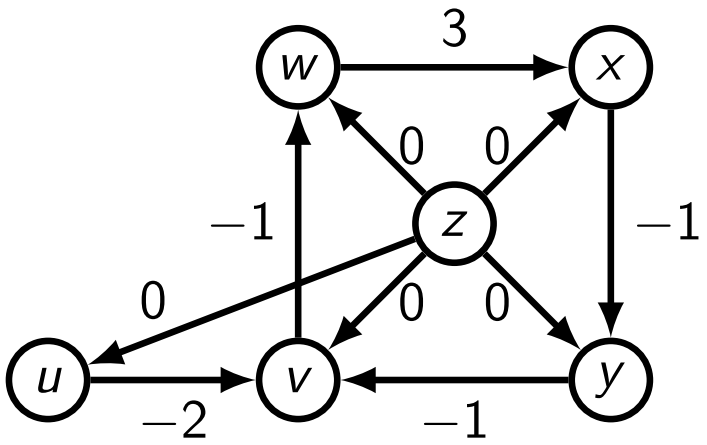
Initial Credit: Outline

1. Determine $X = \{v : \mathrm{E}(v) = 0\}$
   - Canceling non-positive cycles
2. Contract X to a single node z
3. Compute $\mathrm{D}_z(u)$ for every remaining node u

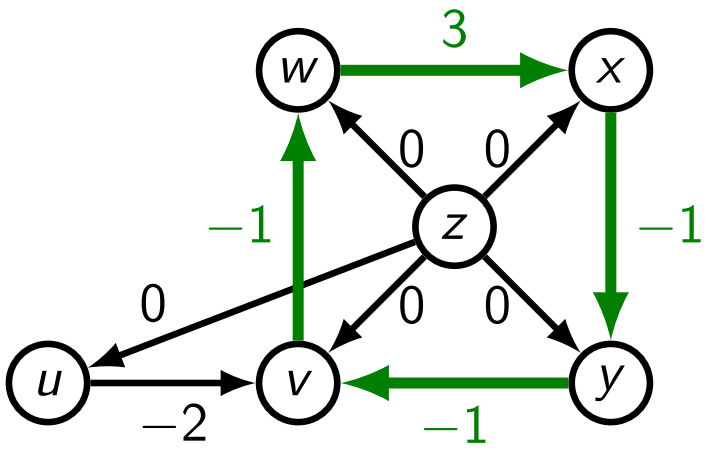
Initial Credit: Example



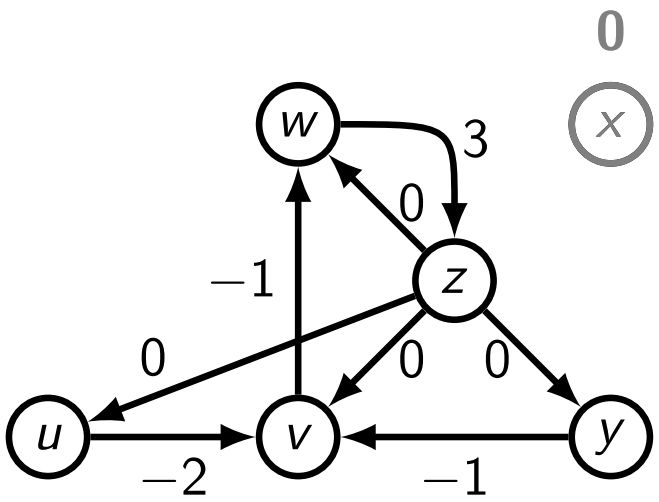
Initial Credit: Example



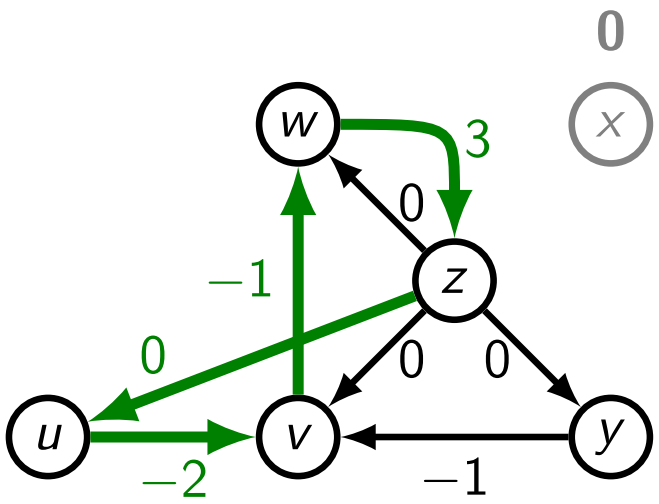
Initial Credit: Example



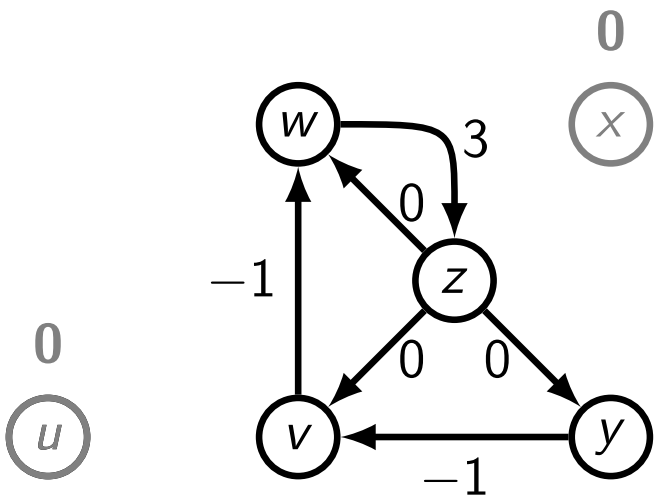
Initial Credit: Example



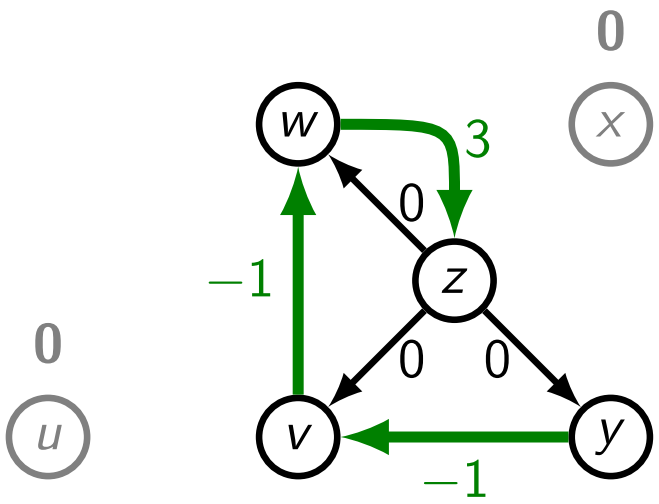
Initial Credit: Example



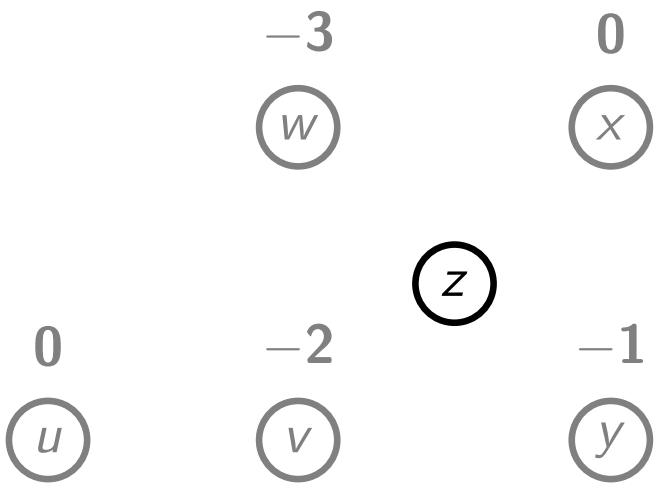
Initial Credit: Example



Initial Credit: Example



Initial Credit: Example



Initial credit

n nodes m edges W max absolute weight

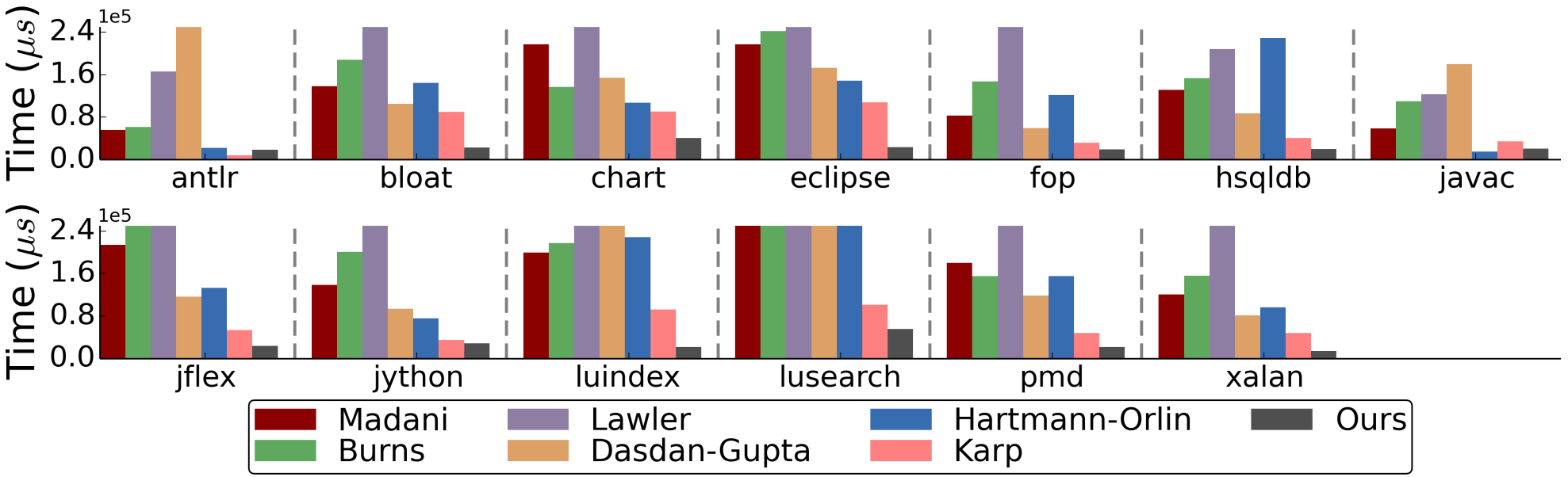
	Existing [Bouyer et al.]	**Our result (general graphs)**	**Our result (constant treewidth)**
Time (decision)	$O(n^2 m)$	$\mathbf{O(nm)}$	$\mathbf{O(n \log n)}$
Time	$O(n^3 m \log(nW))$	$\mathbf{O(\lvert X \rvert nm)}$	$\mathbf{O(n \log n)}$
Space	$O(m)$	$\mathbf{O(m)}$	$\mathbf{O(m) = O(n)}$

Experiments

Experiments

- Mean payoff
  - Control-flow graphs from the DaCapo benchmark suit
  - Assigned random weights in $[-10^3, 10^3]$
  - Compared against 6 existing algorithms with polynomial worst-case guarantees

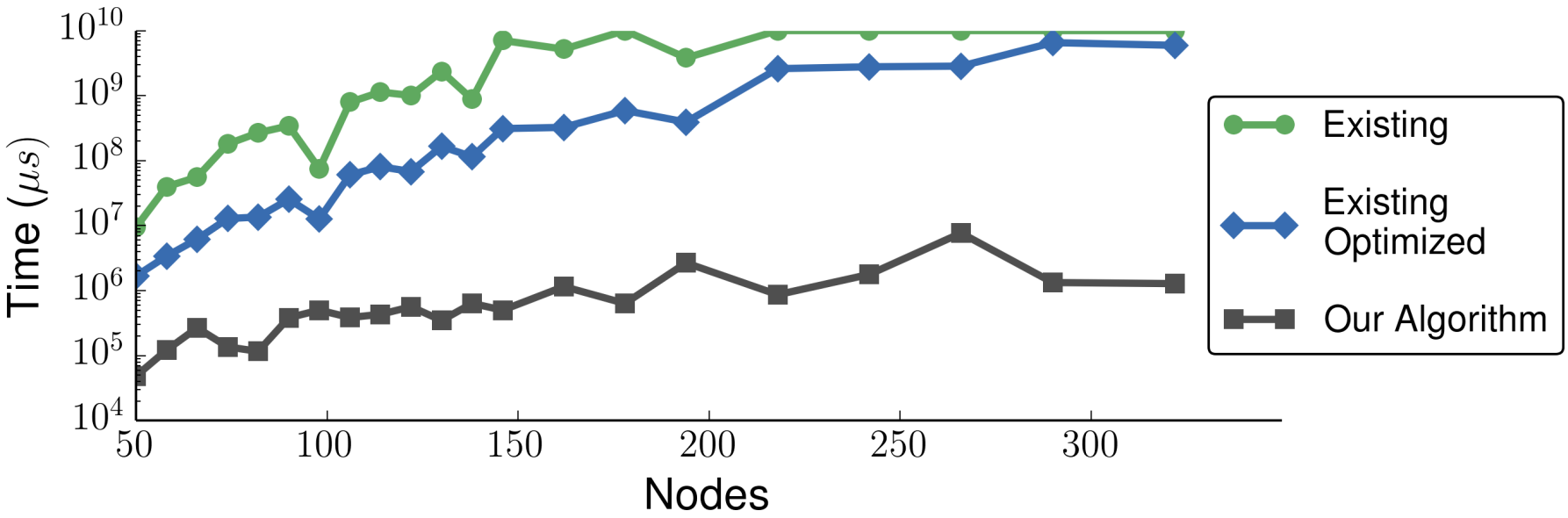
Experimental Results 1: Mean Payoff



Experiments

- Initial credit
  - Weighted graphs from the DIMACS challenge
  - Compared against the existing method [Bouyer et al.] (naive and optimized)

Experimental Results 2: Initial Credit



Thank you!
Questions?