Boolean CSPs and Directed flow-augmentation

Eunjung Kim

CNRS, LAMSADE, Paris-Dauphine University

Joint works with Stefan Kratsch, Marcin Pilipczuk and Magnus Wahlström

17 November 2023, 2nd Workshop on Logic, Graphs and Algorithms, Warsaw, Poland

Min Sat(Γ)

CSPs (Constraint Satisfaction Problems)

- A CSP problem defined by fixing a domain D and a constraint language Γ over D.
- An instance of $\mathrm{CSP}(\Gamma)$ is given as a set of constraints (X, R) over Γ.

Constraint language Γ over a domain D

- a set of relations R over D, each relation $R \subseteq D^r$ for finite r (*arity*).

A constraint (X, R) over a constraint language Γ

- $X = (x_1, \ldots, x_r)$ is an r-tuple of variables (*scope* of the constraint)
- $R \in \Gamma$
- *satisfied* by an assignment $\alpha : X \to D$ if $(\alpha(x_1), \ldots, \alpha(x_r)) \in R$

Min Sat(Γ)

CSPs (Constraint Satisfaction Problems)

- A CSP problem defined by fixing a domain D and a constraint language Γ over D.
- An instance of $\mathrm{CSP}(\Gamma)$ is given as a set of constraints (X, R) over Γ.

Constraint language Γ over a domain D

- a set of relations R over D, each relation $R \subseteq D^r$ for finite r (*arity*).

A constraint (X, R) over a constraint language Γ

- $X = (x_1, \ldots, x_r)$ is an r-tuple of variables (*scope* of the constraint)
- $R \in \Gamma$
- *satisfied* by an assignment $\alpha : X \to D$ if $(\alpha(x_1), \ldots, \alpha(x_r)) \in R$

Min Sat(Γ)

CSPs (Constraint Satisfaction Problems)

- A CSP problem defined by fixing a domain D and a constraint language Γ over D.
- An instance of $\mathrm{CSP}(\Gamma)$ is given as a set of constraints (X, R) over Γ.

Constraint language Γ over a domain D

- a set of relations R over D, each relation $R \subseteq D^r$ for finite r (*arity*).

A constraint (X, R) over a constraint language Γ

- $X = (x_1, \ldots, x_r)$ is an r-tuple of variables (*scope* of the constraint)
- $R \in \Gamma$
- *satisfied* by an assignment $\alpha : X \to D$ if $(\alpha(x_1), \ldots, \alpha(x_r)) \in R$

CSP(Γ), EXAMPLES

	CSP	3-COLORING	2-SAT
Problem	domain D	$\{1,2,3\}$	$\{0,1\}$
	constraint language Γ, $R \subseteq D^r$ for each $R \in \Gamma$ of arity r	$\{\neq\}$	$\{\{0,1\}^2 \setminus (a,b) : a,b \in \{0,1\}\}$
Instance	variables V	vertices	variables
	constraints (X,R), $R \in \Gamma$, $X \in V^r$	$\{(u,v),\neq)\}_{uv \in E(G)}$	clauses

We focus on boolean constraint languages, where CSP(Γ) is now called SAT(Γ).

$\mathrm{CSP}(\Gamma)$, EXAMPLES

	CSP	3-COLORING	2-SAT
Problem	domain D	$\{1,2,3\}$	$\{0,1\}$
	constraint language Γ, $R \subseteq D^r$ for each $R \in \Gamma$ of arity r	$\{\neq\}$	$\{\{0,1\}^2 \setminus (a,b) : a,b \in \{0,1\}\}$
Instance	variables V	vertices	variables
	constraints (X,R), $R \in \Gamma$, $X \in V^r$	$\{(u,v),\neq)\}_{uv \in E(G)}$	clauses

We focus on boolean constraint languages, where $\mathrm{CSP}(\Gamma)$ is now called $\mathrm{SAT}(\Gamma)$.

A natural variant of Sat(Γ): Min Sat(Γ)

Min Sat(Γ)

Input: A formula (i.e. a set of constraints) $\mathcal{F}$ over Γ, where the domain of Γ is boolean, a non-negative integer k.

Question: is there a set of at most k constraints $Z \subseteq \mathcal{F}$ such that $\mathcal{F} - Z$ is satisfiable?

st-Min-Cut as Min Sat(Γ)

Alternative formulation of st-Min-Cut: Given $G = (V, E)$ with $s, t \in V$ and integer k,

- Variables V.
- $(s = 1)$ and $(t = 0)$ as *crisp* constraints (i.e. $k + 1$ copies).
- For every edge $e = (s, v) \in E$, a constraint $(v = 1)$.
- For every edge $e = (u, t) \in E$, a constraint $(u = 0)$.
- For every other edge $e = (u, v) \in E$, a constraint $(u = v)$.
- Find a boolean assignment $\phi : V \to \{0, 1\}$ such that all but at most k constraints are satisfied.

This is a formula of Min Sat(Γ) over $\Gamma = \{0, 1, =\}$.

st-Min-Cut as Min Sat(Γ)

Alternative formulation of st-Min-Cut: Given $G = (V, E)$ with $s, t \in V$ and integer k,

- Variables V.
- $(s = 1)$ and $(t = 0)$ as *crisp* constraints (i.e. $k + 1$ copies).
- For every edge $e = (s, v) \in E$, a constraint $(v = 1)$.
- For every edge $e = (u, t) \in E$, a constraint $(u = 0)$.
- For every other edge $e = (u, v) \in E$, a constraint $(u = v)$.
- Find a boolean assignment $\phi : V \to \{0, 1\}$ such that all but at most k constraints are satisfied.

This is a formula of Min Sat(Γ) over $\Gamma = \{0, 1, =\}$.

st-Min-Cut as Min Sat(Γ)

Alternative formulation of st-Min-Cut: Given $G = (V, E)$ with $s, t \in V$ and integer k,

- Variables V.
- $(s = 1)$ and $(t = 0)$ as *crisp* constraints (i.e. $k + 1$ copies).
- For every edge $e = (s, v) \in E$, a constraint $(v = 1)$.
- For every edge $e = (u, t) \in E$, a constraint $(u = 0)$.
- For every other edge $e = (u, v) \in E$, a constraint $(u = v)$.
- Find a boolean assignment $\phi : V \to \{0, 1\}$ such that all but at most k constraints are satisfied.

This is a formula of Min Sat(Γ) over $\Gamma = \{0, 1, =\}$.

st-Min-Cut as Min Sat(Γ)

Alternative formulation of st-Min-Cut: Given $G = (V, E)$ with $s, t \in V$ and integer k,

- Variables V.
- $(s = 1)$ and $(t = 0)$ as *crisp* constraints (i.e. $k + 1$ copies).
- For every edge $e = (s, v) \in E$, a constraint $(v = 1)$.
- For every edge $e = (u, t) \in E$, a constraint $(u = 0)$.
- For every other edge $e = (u, v) \in E$, a constraint $(u = v)$.
- Find a boolean assignment $\phi : V \to \{0, 1\}$ such that all but at most k constraints are satisfied.

This is a formula of Min Sat(Γ) over $\Gamma = \{0, 1, =\}$.

st-Min-Cut as Min Sat(Γ)

Alternative formulation of st-Min-Cut: Given $G = (V, E)$ with $s, t \in V$ and integer k,

- Variables V.
- $(s = 1)$ and $(t = 0)$ as *crisp* constraints (i.e. $k + 1$ copies).
- For every edge $e = (s, v) \in E$, a constraint $(v = 1)$.
- For every edge $e = (u, t) \in E$, a constraint $(u = 0)$.
- For every other edge $e = (u, v) \in E$, a constraint $(u = v)$.
- Find a boolean assignment $\phi : V \to \{0, 1\}$ such that all but at most k constraints are satisfied.

This is a formula of Min Sat(Γ) over $\Gamma = \{0, 1, =\}$.

st-Min-Cut as Min Sat(Γ)

Alternative formulation of st-Min-Cut: Given $G = (V, E)$ with $s, t \in V$ and integer k,

- Variables V.
- $(s = 1)$ and $(t = 0)$ as *crisp* constraints (i.e. $k + 1$ copies).
- For every edge $e = (s, v) \in E$, a constraint $(v = 1)$.
- For every edge $e = (u, t) \in E$, a constraint $(u = 0)$.
- For every other edge $e = (u, v) \in E$, a constraint $(u = v)$.
- Find a boolean assignment $\phi : V \to \{0, 1\}$ such that all but at most k constraints are satisfied.

This is a formula of Min Sat(Γ) over $\Gamma = \{0, 1, =\}$.

st-Min-Cut as Min Sat(Γ)

Alternative formulation of *st*-Min-Cut: Given $G = (V, E)$ with $s, t \in V$ and integer k,

- Variables V.
- $(s = 1)$ and $(t = 0)$ as *crisp* constraints (i.e. $k + 1$ copies).
- For every edge $e = (s, v) \in E$, a constraint $(v = 1)$.
- For every edge $e = (u, t) \in E$, a constraint $(u = 0)$.
- For every other edge $e = (u, v) \in E$, a constraint $(u = v)$.
- Find a boolean assignment $\phi : V \to \{0, 1\}$ such that all but at most k constraints are satisfied.

This is a formula of Min Sat(Γ) over $\Gamma = \{0, 1, =\}$.

st-Min-Cut as Min Sat(Γ)

Alternative formulation of *st*-Min-Cut: Given $G = (V, E)$ with $s, t \in V$ and integer k,

- Variables V.
- $(s = 1)$ and $(t = 0)$ as *crisp* constraints (i.e. $k + 1$ copies).
- For every edge $e = (s, v) \in E$, a constraint $(v = 1)$.
- For every edge $e = (u, t) \in E$, a constraint $(u = 0)$.
- For every other edge $e = (u, v) \in E$, a constraint $(u = v)$.
- Find a boolean assignment $\phi : V \to \{0, 1\}$ such that all but at most k constraints are satisfied.

This is a formula of Min Sat(Γ) over $\Gamma = \{0, 1, =\}$.

k-Clique as Min Sat(Γ)

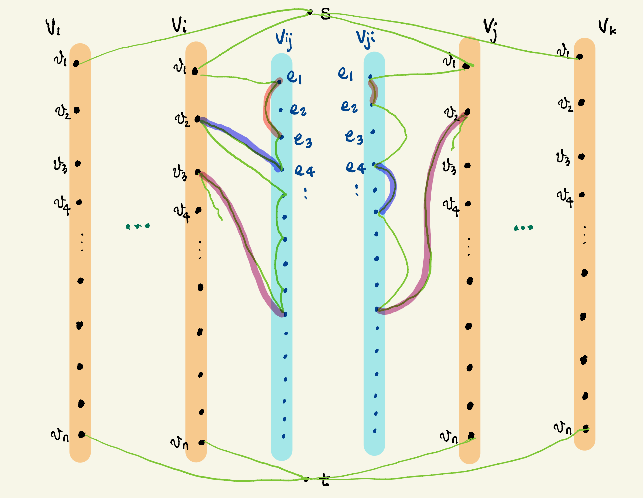
k-Multicolored Clique (equivalent to k-Clique):

- Input: $G = (V_1 \uplus V_2 \uplus \cdots V_k, E)$ with each V_i stable, integer k.
- Task: find a k-clique K (report No if none exists).

Alternative formulation as Min Sat(Γ) with $\Gamma = \{=, 0, 1, \text{double equality}\}$
(Marx and Razgon 2009)

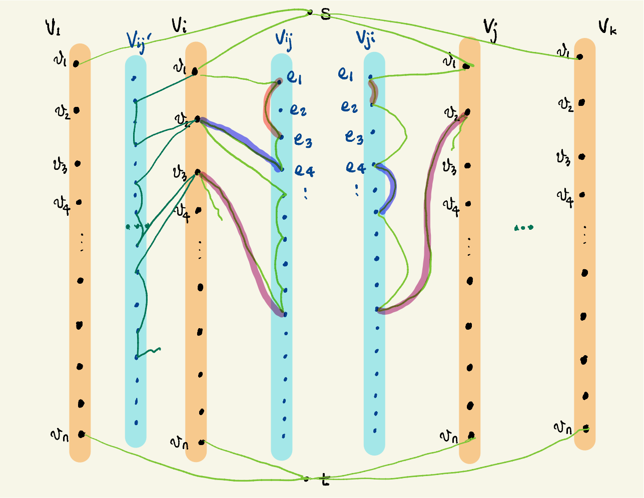
k-Clique as Min Sat(Γ)

Does the following instance of Min Sat(Γ), where $\Gamma = \{=, 0, 1, \text{DOUBLE EQUALITY}\}$, admit a boolean assignment violating at most $\binom{k}{2}$ constraints?



k-Clique as Min Sat(Γ)

Does the following instance of Min Sat(Γ), where $\Gamma = \{=, 0, 1, \text{DOUBLE EQUALITY}\}$, admit a boolean assignment violating at most $\binom{k}{2}$ constraints?



Toward FPT dichotomy for Min Sat(Γ)

<table>
<tr><th colspan="2">constraint type</th><th>Feasibility in P-time (Schaefer'78)</th><th>c-approx in FPT-time (BELM'18)</th><th>FPT</th></tr>
<tr><td colspan="2">0/1-valid</td><td colspan="3">trivially satisfiable</td></tr>
<tr><td colspan="2">bijunctive (2CNF)</td><td rowspan="4">Yes</td><td rowspan="2">Yes</td><td rowspan="2">Something interesting happens here</td></tr>
<tr><td rowspan="2">weakly neg/pos (horn/dual horn)</td><td>IHS-B</td></tr>
<tr><td>o/w</td><td rowspan="2">No</td><td rowspan="2">No</td></tr>
<tr><td colspan="2">affine</td></tr>
<tr><td colspan="2">o/w</td><td colspan="3">No</td></tr>
</table>

FPT dichotomy for Min Sat(Γ)

K. Kratsch, Pilipczuk, Wahlström 2021,22,23

Theorem

Let Γ be a finite boolean constraint language. Then parameterize by the number of unsatisfied constraints, one of the following holds.

1. Weighted Min Sat(Γ) *is FPT.*
2. Min Sat(Γ) *is FPT, but* Weighted Min Sat(Γ) *is W[1]-hard.*
3. Min Sat(Γ) *is W[1]-hard.*

The hard gist of the tractable cases critically relay on the flow-augmentation technique.

FPT dichotomy for Min Sat(Γ)

K. Kratsch, Pilipczuk, Wahlström 2021,22,23

Theorem

Let Γ be a finite boolean constraint language. Then parameterize by the number of unsatisfied constraints, one of the following holds.

1. Weighted Min Sat(Γ) *is FPT.*
2. Min Sat(Γ) *is FPT, but* Weighted Min Sat(Γ) *is W[1]-hard.*
3. Min Sat(Γ) *is W[1]-hard.*

The hard gist of the tractable cases critically relay on the flow-augmentation technique.

FPT dichotomy for Min Sat(Γ)

K. Kratsch, Pilipczuk, Wahlström 2021,22,23

<table>
<tr><th colspan="3">constraint type</th><th>Feasibility in P-time (S'78)</th><th>c-approx in FPT-time (BELM'18)</th><th>FPT</th></tr>
<tr><td colspan="3">0/1-valid</td><td colspan="3">trivially satisfiable</td></tr>
<tr><td colspan="2" rowspan="2">bijunctive (2CNF)</td><td>$2K_2$-free Gaifman graph</td><td rowspan="6">Yes</td><td rowspan="4">Yes</td><td>Yes (even weighted)</td></tr>
<tr><td>o/w</td><td>No</td></tr>
<tr><td rowspan="3">weakly neg/pos (horn/ dual horn)</td><td rowspan="2">IHS-B</td><td>$2K_2$-free arrow graph</td><td>Yes</td></tr>
<tr><td>o/w</td><td>No</td></tr>
<tr><td colspan="2">o/w</td><td rowspan="2">No</td><td rowspan="2">No</td></tr>
<tr><td colspan="3">affine</td></tr>
<tr><td colspan="3">o/w</td><td colspan="3">No</td></tr>
</table>

Example of a tractable case: ℓ-Chain Sat

ℓ-Chain Sat

- Input: a formula Φ as a set of constraints of the following form.
  - $(x_1 \rightarrow x_2) \wedge (x_2 \rightarrow x_3) \wedge \cdots (x_{\ell-1} \rightarrow x_\ell)$,
  - or unary clauses, i.e. (x) or $(\neg x)$

  each constraint C has weight $\omega(C)$. Two integers k and W.
- Task: find a truth assignment $V(\Phi) \rightarrow \{1, 0\}$ violating at most k constraints of weight at most W.

$\Phi = \bigwedge_{i=1}^{4}(x_i) \wedge \bigwedge_{i=1}^{4}(\neg y_i) \wedge \bigwedge_{i=1}^{4} C_i$,

$C_i = (x_i \rightarrow w_i) \wedge (w_i \rightarrow y_i) \wedge (y_i \rightarrow w_{i+1})$

FPT in $k + \ell$? Was THE bottleneck cut problem.
Tractable case in both bijunctive and IHS-B languages.
$\ell = 1, 2$ is st-Min-Cut.

Example of a tractable case: ℓ-Chain Sat

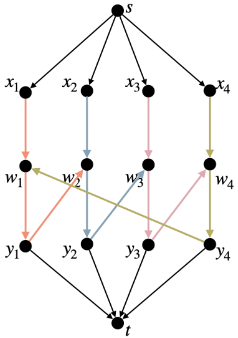
ℓ-Chain Sat

- Input: a formula Φ as a set of constraints of the following form.
  - $(x_1 \to x_2) \wedge (x_2 \to x_3) \wedge \cdots (x_{\ell-1} \to x_\ell)$,
  - or unary clauses, i.e. (x) or $(\neg x)$

  each constraint C has weight $\omega(C)$. Two integers k and W.
- Task: find a truth assignment $V(\Phi) \to \{1, 0\}$ violating at most k constraints of weight at most W.

$\Phi = \bigwedge_{i=1}^{4}(x_i) \wedge \bigwedge_{i=1}^{4}(\neg y_i) \wedge \bigwedge_{i=1}^{4} C_i$,

$C_i = (x_i \to w_i) \wedge (w_i \to y_i) \wedge (y_i \to w_{i+1})$



FPT in $k + \ell$? Was THE bottleneck cut problem.
Tractable case in both bijunctive and IHS-B languages.
$\ell = 1, 2$ is st-Min-Cut.

Example of a tractable case: ℓ-Chain Sat

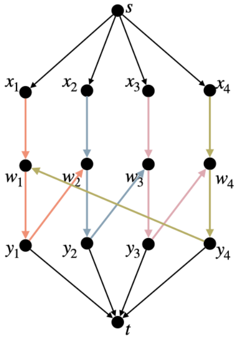
ℓ-Chain Sat

- Input: a formula Φ as a set of constraints of the following form.
  - $(x_1 \to x_2) \wedge (x_2 \to x_3) \wedge \cdots (x_{\ell-1} \to x_\ell)$,
  - or unary clauses, i.e. (x) or $(\neg x)$

  each constraint C has weight $\omega(C)$. Two integers k and W.
- Task: find a truth assignment $V(\Phi) \to \{1, 0\}$ violating at most k constraints of weight at most W.

$\Phi = \bigwedge_{i=1}^{4}(x_i) \wedge \bigwedge_{i=1}^{4}(\neg y_i) \wedge \bigwedge_{i=1}^{4} C_i$,

$C_i = (x_i \to w_i) \wedge (w_i \to y_i) \wedge (y_i \to w_{i+1})$



FPT in $k + \ell$? Was THE bottleneck cut problem.

Tractable case in both bijunctive and IHS-B languages.

$\ell = 1, 2$ is st-Min-Cut.

Example of a tractable case: ℓ-Chain Sat

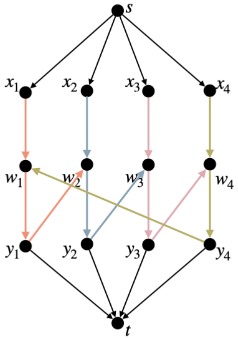
ℓ-Chain Sat

- Input: a formula Φ as a set of constraints of the following form.
  - $(x_1 \to x_2) \wedge (x_2 \to x_3) \wedge \cdots (x_{\ell-1} \to x_\ell)$,
  - or unary clauses, i.e. (x) or $(\neg x)$

  each constraint C has weight $\omega(C)$. Two integers k and W.
- Task: find a truth assignment $V(\Phi) \to \{1, 0\}$ violating at most k constraints of weight at most W.

$\Phi = \bigwedge_{i=1}^{4}(x_i) \wedge \bigwedge_{i=1}^{4}(\neg y_i) \wedge \bigwedge_{i=1}^{4} C_i$,

$C_i = (x_i \to w_i) \wedge (w_i \to y_i) \wedge (y_i \to w_{i+1})$



FPT in $k + \ell$? Was THE bottleneck cut problem.
Tractable case in both bijunctive and IHS-B languages.
$\ell = 1, 2$ is st-Min-Cut.

Example of a tractable case: ℓ-Chain Sat

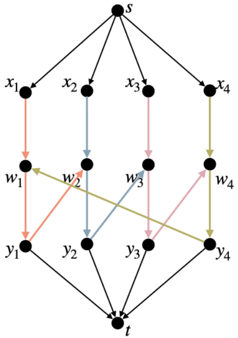
ℓ-Chain Sat

- Input: a formula Φ as a set of constraints of the following form.
  - $(x_1 \to x_2) \wedge (x_2 \to x_3) \wedge \cdots (x_{\ell-1} \to x_\ell)$,
  - or unary clauses, i.e. (x) or $(\neg x)$

  each constraint C has weight $\omega(C)$. Two integers k and W.
- Task: find a truth assignment $V(\Phi) \to \{1, 0\}$ violating at most k constraints of weight at most W.

$\Phi = \bigwedge_{i=1}^{4}(x_i) \wedge \bigwedge_{i=1}^{4}(\neg y_i) \wedge \bigwedge_{i=1}^{4} C_i,$

$C_i = (x_i \to w_i) \wedge (w_i \to y_i) \wedge (y_i \to w_{i+1})$



FPT in $k+\ell$? Was THE bottleneck cut problem.
Tractable case in both bijunctive and IHS-B languages.
$\ell = 1, 2$ is st-Min-Cut.

Flow augmentation

Flow Augmentation Theorem (simple version)

There exists a polynomial-time algorithm that, given

- a directed graph G with $s, t \in V(G)$ and an integer k,

returns

- a set $A \subseteq V(G) \times V(G)$

such that for every minimal st-cut Z of size at most k, with probability $2^{-\mathcal{O}(k^4 \log k)}$

- Z is an st-cut of minimum cardinality in $G + A$.

When a sought solution Z is a minimal st-cut, then flow-augmentation lifts st-mincut size to match $|Z|$ by adding (unbreakable) arcs in a way not messing the solution, with high enough probability.

Flow augmentation

Flow Augmentation Theorem (simple version)

There exists a polynomial-time algorithm that, given

- a directed graph G with $s, t \in V(G)$ and an integer k,

returns

- a set $A \subseteq V(G) \times V(G)$

such that for every minimal st-cut Z of size at most k, with probability $2^{-\mathcal{O}(k^4 \log k)}$

- Z is an st-cut of minimum cardinality in $G + A$.

When a sought solution Z is a minimal st-cut, then flow-augmentation lifts st-mincut size to match $|Z|$ by adding (unbreakable) arcs in a way not messing the solution, with high enough probability.

Algorithm for ℓ-Chain Sat

Weighted ℓ-Chain Sat

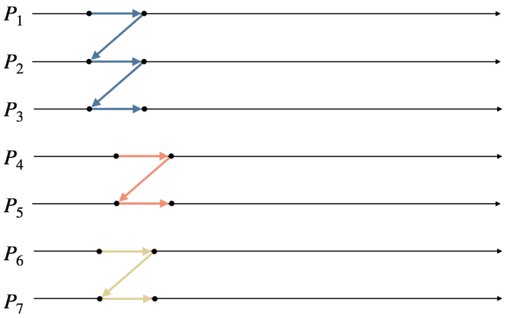
- Input: a directed graph $G = (V, E)$ with s, t, a collection $\mathcal{B}$ (bundles) of pairwise disjoint path of length at most b with weights $\omega : \mathcal{B} \to \mathbb{Z}_+$, integers k and W.
- Task: find a minimal st-cut $Z \subseteq \bigcup \mathcal{B}$ violating at most k bundles of weight at most W.

1. Invoke flow-augmentation; now Z is an st-mincut. Note $\lambda(s, t) \leq bk$

Algorithm for ℓ-Chain Sat

Weighted ℓ-Chain Sat

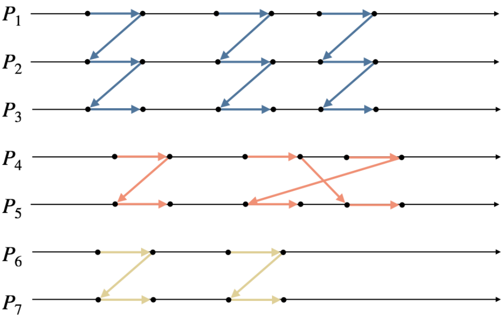
- Input: a directed graph $G = (V, E)$ with s, t, a collection $\mathcal{B}$ (bundles) of pairwise disjoint path of length at most b with weights $\omega : \mathcal{B} \to \mathbb{Z}_+$, integers k and W.
- Task: find a minimal st-cut $Z \subseteq \bigcup \mathcal{B}$ violating at most k bundles of weight at most W.



2. Guess how violated bundles overlay the flow paths.

Weighted ℓ-Chain Sat

- Input: a directed graph $G = (V, E)$ with s, t, a collection $\mathcal{B}$ (bundles) of pairwise disjoint path of length at most b with weights $\omega : \mathcal{B} \rightarrow \mathbb{Z}_+$, integers k and W.
- Task: find a minimal st-cut $Z \subseteq \bigcup \mathcal{B}$ violating at most k bundles of weight at most W.

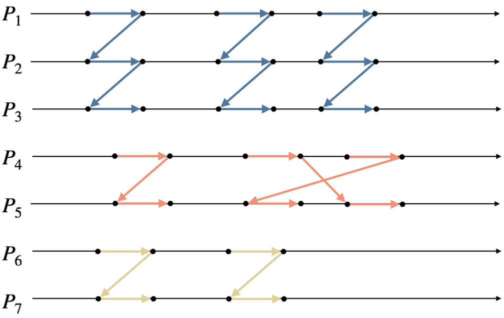


3. Consider (only) the bundles conforming the guess.

Algorithm for ℓ-Chain Sat

Weighted ℓ-Chain Sat

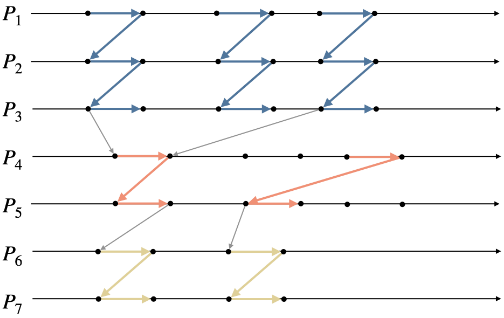
- Input: a directed graph $G = (V, E)$ with s, t, a collection $\mathcal{B}$ (bundles) of pairwise disjoint path of length at most b with weights $\omega : \mathcal{B} \to \mathbb{Z}_+$, integers k and W.
- Task: find a minimal st-cut $Z \subseteq \bigcup \mathcal{B}$ violating at most k bundles of weight at most W.



4. A bundle *crossed* by another bundle cannot be violated.

WEIGHTED ℓ-CHAIN SAT

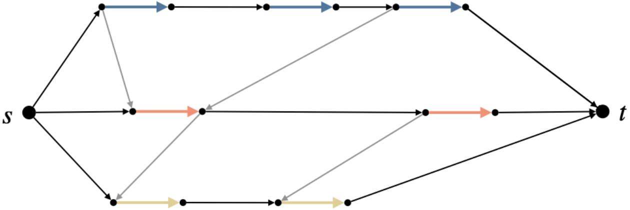
- Input: a directed graph $G = (V, E)$ with s, t, a collection $\mathcal{B}$ (bundles) of pairwise disjoint path of length at most b with weights $\omega : \mathcal{B} \to \mathbb{Z}_+$, integers k and W.
- Task: find a minimal st-cut $Z \subseteq \bigcup \mathcal{B}$ violating at most k bundles of weight at most W.



5. Bundles are *linearly ordered*.

WEIGHTED ℓ-CHAIN SAT

- Input: a directed graph $G = (V, E)$ with s, t, a collection $\mathcal{B}$ (bundles) of pairwise disjoint path of length at most b with weights $\omega : \mathcal{B} \to \mathbb{Z}_+$, integers k and W.
- Task: find a minimal st-cut $Z \subseteq \bigcup \mathcal{B}$ violating at most k bundles of weight at most W.



6. *Compress* the bundles, obtain (Weighted) st-MINCUT instance.

Concluding remarks

- $2K_2$-freeness allows one to use a similar argument in the more general cases.
- The presence of $2K_2$ leads to a reduction in the spirit of the previous one (k-Clique to Min Sat($\{=, 0, 1, \text{double equality}\}$).
- *Flow-augmentation* looks like the missing tool in directed graph separation problems.
- It closed some dichotomies and long-standing open problems.
- Key open problemArgh...it's closed recently by George Osipov and Marcin Pilipczuk: Symmetric Multicut.
  Directed graph G, unordered pairs of terminals $\mathcal{T}$, integer k. Delete k edges so that for every $st \in \mathcal{T}$, s and t are not in the same strong component.