# Solving hard cut problems via flow augmentation

Joint work with Eunjung Kim , Stefan Kratsch, Marcin Pilipczuk

Warwick, 27 October 2020

ROYAL HOLLOWAY UNIVERSITY OF LONDON

# Overview

1. Flow augmentation
2. Application: Min CSP characterization
3. Application: Coupled Min-Cut

# Cut problems

- Tractable…
  - MIN (S,T)-CUT
- …and intractable
  - MULTIWAY CUT – separate at least 3 terminals from each other
  - MULTICUT – fulfil arbitrary set of cut requests $(s_i, t_i)$
  - BISECTION – find a min-cut with n/2 vertices per side
  - …

# FPT algorithms

- A **parameterized problem** is a problem where every input is given with a parameter *k* (e.g., solution size)
- A problem is **Fixed-Parameter Tractable (FPT)** parameterized by *k* if instances (for some function *f(k)*) can be solved in time

$$f(k) \cdot n^{O(1)}$$

- The contrast (intractable) is running times like $n^k$ or $2^n$

# FPT algorithms for cut problems

- MULTIWAY CUT: $2^{O(k)} \cdot O(m)$ time
  - Marx [06]; Iwata, Oka, Yoshida [14]; Iwata, Yamagochi [18] – various methods
- MULTICUT: $2^{poly(k)} \cdot n^{O(1)}$ time
  - Bousquet, Daligault, Thomassé [11/18]; Marx, Razgon [11/14] – various methods
- **Frameworks** for cut problems
  - **Treewidth reduction** [Marx, O'Sullivan, Razgon 13]
  - **Complete graph decomposition** [CLPPS 14/19; CKLPPSW 20+]

# Our sample problems

ROYAL HOLLOWAY UNIVERSITY OF LONDON

1. **Bi-Objective (s,t)-Cut**
   - Given $G = (V, E)$, vertices $s, t$, integers $k, W$
   - Find $(s, t)$-cut $Z \subseteq E$ such that
     1. $|Z| \leq k$
     2. $w(Z) \leq W$ (according to edge weights $w(e)$)
2. **Coupled Min-Cut** (defined later)

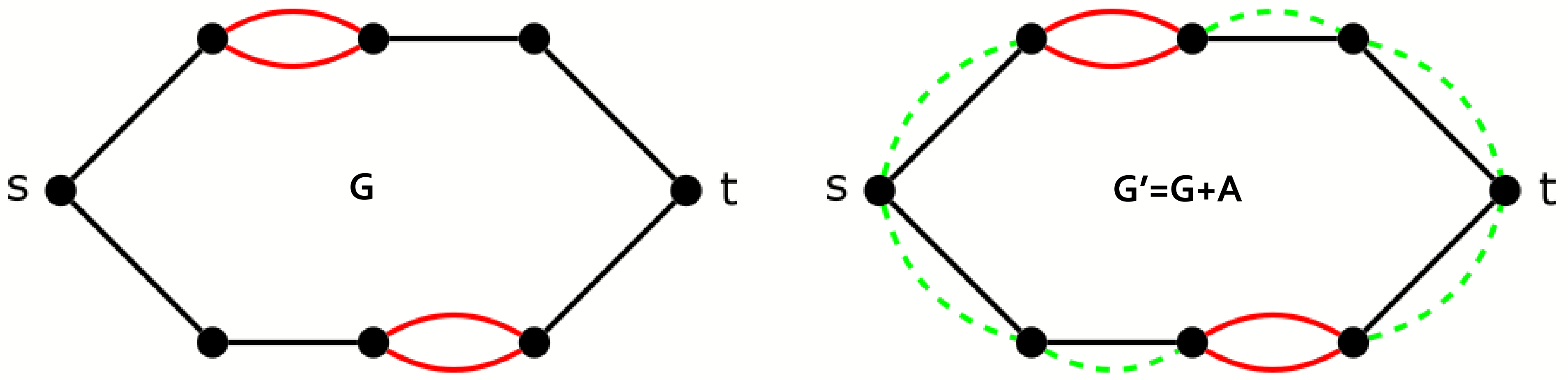
# Bi-Objective (s,t)-Cut

- NP-hard in general [Papadimitriou, Yannakakis 01]
- Tractable if solution is min-cut:
  - Set new edge weights $w'(e) = w(e) + W$
  - Compute (s,t)-min cut with weights $w'$
  - Cut budget $kW + W$
- Reduction fails if $\lambda_G(s,t) < k$

# Flow Augmentation

Given $G = (V, E)$ with unknown minimal $(s, t)$-cut $Z \subseteq E$, and parameter k=$|Z|$, we can compute augmented graph $G' = G + A$ such that

1. Construction of $G'$ takes $k^{O(1)} \cdot O(m)$ time
2. With probability at least one in $2^{O(k \log k)}$, $Z$ is **(s,t)-min cut** in $G'$



# Bi-objective (s,t)-cut

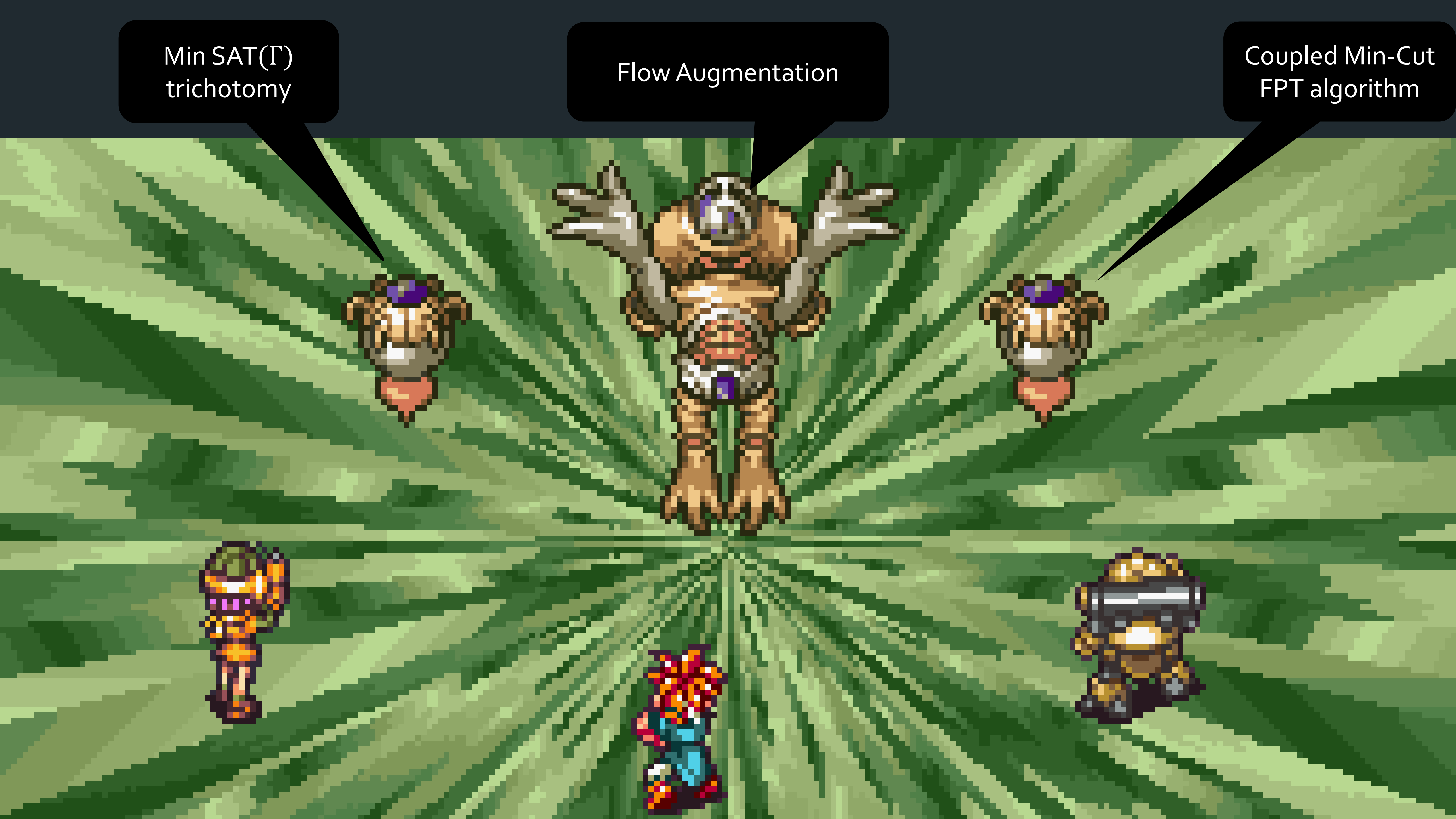
- FPT algorithm for Bi-objective (s,t)-cut: [Time $2^{O(k \log k)} \cdot O(m \log n)$]

1. Repeat $2^{\Theta(k \log k)}$ times:
   - Compute augmentation G'=G+A with target flow $\lambda_{G+A}(s,t) = k$
   - Solve problem as if Z is min-cut in G'
2. Return cheapest solution found

This is a toy problem – but this is a competitive running time

# Results overview

ROYAL HOLLOWAY UNIVERSITY OF LONDON


Min SAT(Γ) trichotomy
Flow Augmentation
Coupled Min-Cut FPT algorithm

# Min SAT problem family



**Min SAT($\Gamma$) with constraint language $\Gamma$:**

- **Input:** Formula $F = R_1(X_1) \wedge R_2(X_2) \wedge \cdots$ of constraints $R_i \in \Gamma$; integer $k$
- **Question:** Is there an assignment to $F$ with at most $k$ false constraints $R_i(X_i)$?

Examples:

- **Undirected (s,t)-Cut:** Language $\Gamma = \{0, 1, (x = y)\}$:
  - $F = (s = 1) \wedge (s = v_1) \wedge (v_1 = v_2) \wedge \cdots \wedge (t = 0)$
- **Edge Bipartization:** Language $\Gamma = \{(x \neq y)\}$
- **Almost 2-SAT:** Language $\Gamma = \{(x \vee y), (x \vee \overline{y}), (\overline{x} \vee \overline{y})\}$
- **$\ell$-Chain SAT:** $\Gamma = \{0, 1, (x_0 \to x_1 \to \cdots \to x_\ell)\}$

# Min SAT characterization result

For every Boolean language $\Gamma$, one of the following holds:

1. Min SAT($\Gamma$) is **FPT** (using flow augmentation)
2. Min SAT($\Gamma$) is **W[1]-hard**
3. Min SAT($\Gamma$) encompasses **directed cut problems** (such as $\ell$-Chain SAT)

Define two relations:

- $R_4(a,b,c,d) \equiv (a=b) \wedge (c=d) \wedge (\overline{a} \vee \overline{c})$
- $R_{hard}(a,b,c,d) \equiv (a=b) \wedge (c=d)$

Then:

- **Min SAT($\Gamma$)** with $\Gamma = \{0,1,R_4\}$ defines **Coupled Min-Cut** and is **FPT**
- **Min SAT($\Gamma$)** with $\Gamma = \{0,1,R_{hard}\}$ is **W[1]-hard** ***(double equality)***

# Min SAT cases – further details

**Min SAT($\Gamma$)** falls into one of the following cases (up to duality):

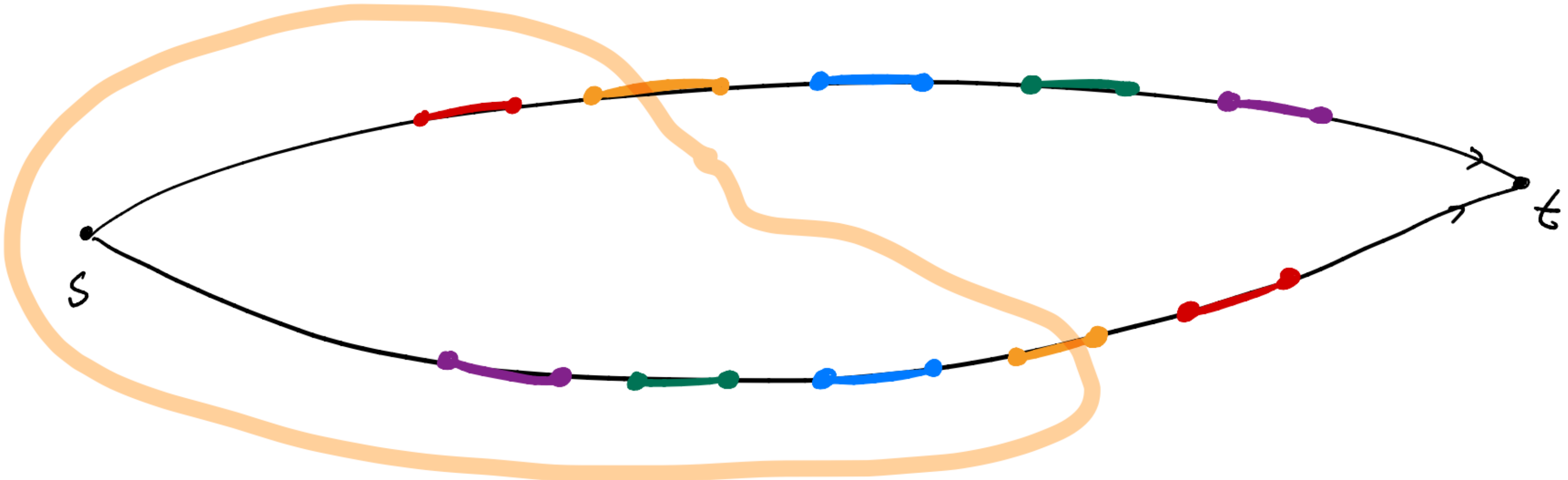
1. Reduces to relation $(a = 0) \land (b = 1) \land (c \neq d)$ [Edge Bipartization]
2. Reduces to $(a = 1) \land (c = d)$ and $(\overline{x}_1 \lor \cdots \lor \overline{x}_d)$ [Graph Pair Cut, etc.]
3. **Generalized Coupled Min-Cut (GCMC)**
4. W[1]-hard
5. Implements directed graph cut problems

Cases 1-3 FPT, cases 1-2 use standard methods.

# Coupled Min-Cut

Graph formulation:

- Graph $G = (V, E)$, vertices $s, t$, budget bound $k$
- Looking for $(s, t)$-cut $Z$ of cost at most $k$
- Edges of $E$ are grouped into **pairs** $(e, f)$ where
  1. Edges $e, f$ can be cut simultaneously at cost 1, but
  2. if edges $e, f$ **not** cut then at most one edge is on the $s$-side of the cut



# Coupled Min-Cut properties



- NP-hard in general
- Tractable (FPT) if $Z$ is min-cut
- Edge Multicut $\leq_{FPT}$ Coupled Min-Cut
- Impervious (?) to previous methods
  - Important separators, shadow removal, decomposition methods don't apply
  - LP duality???

# Flow Augmentation overview

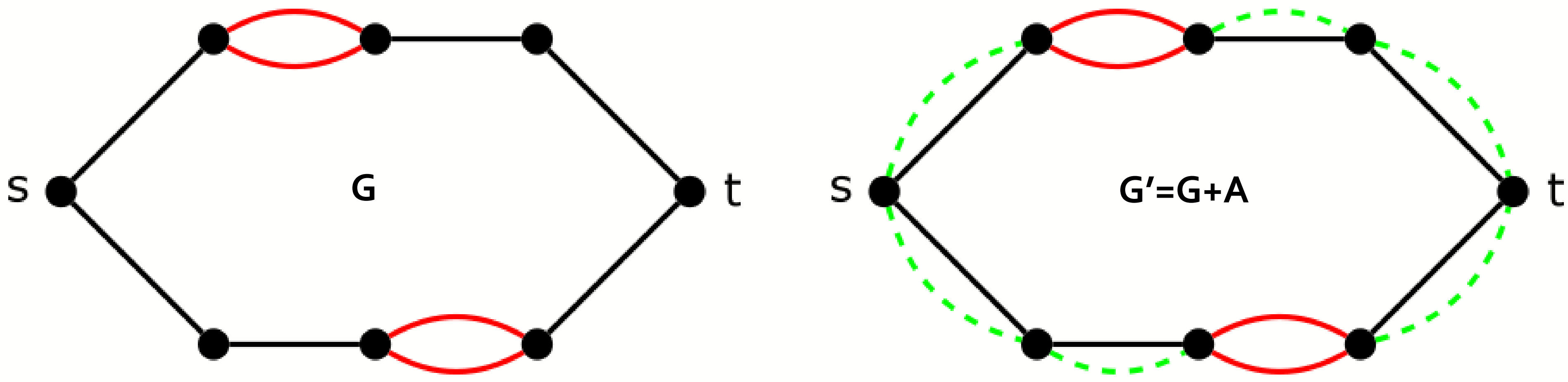
ROYAL HOLLOWAY UNIVERSITY OF LONDON

# Flow Augmentation

- Graph $G = (V, E)$, vertices $s, t$, target flow $k$
- Unknown minimal $(s,t)$-cut $Z$, $|Z| = k$

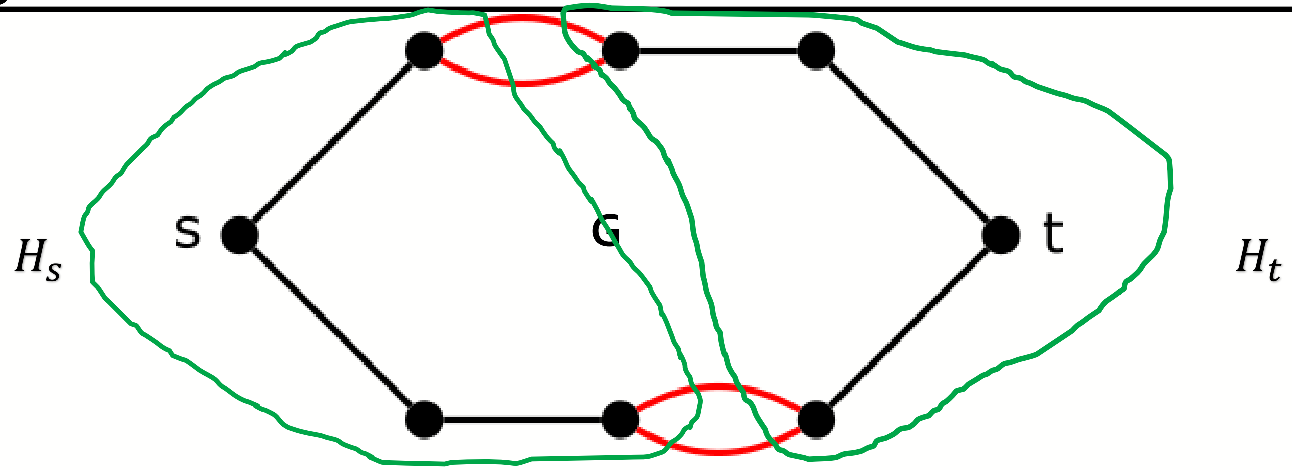
**Task:**

- Add edges $A$ to $G$ so that $Z$ is $(s,t)$-min cut in $G + A$



# Flow Augmentation observations

1. $G - Z$ partitions into two components $H_s, H_t$
2. Adding $(u, v)$ to $A$ **forbidden** if and only if $u \in H_s, v \in H_t$
3. Suffices to:
   a. Add edges to $A$ increasing $(s, t)$-max-flow, such that
   b. No such edge is **forbidden**



# A strategy

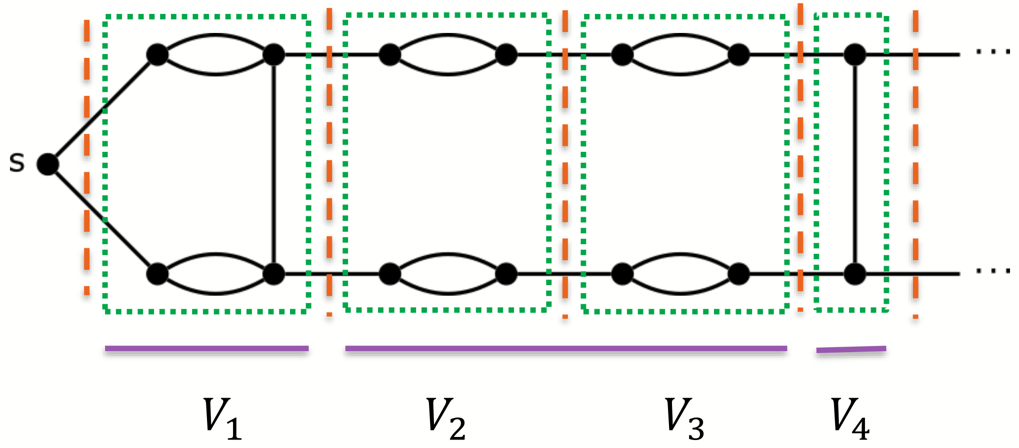
- Trace "all" min-cuts from $s$ to $t$
- Guess how $Z$ interacts with each one
- Add augmenting edges consistent with the guess

Reasons it could work:

- If a cut $C$ is entirely in $H_s$ (or $H_t$), there are no bad guesses
- If we can control "mixed" cuts $C$ intersecting both components, there may be only $f(k)$ places where we need to guess correctly

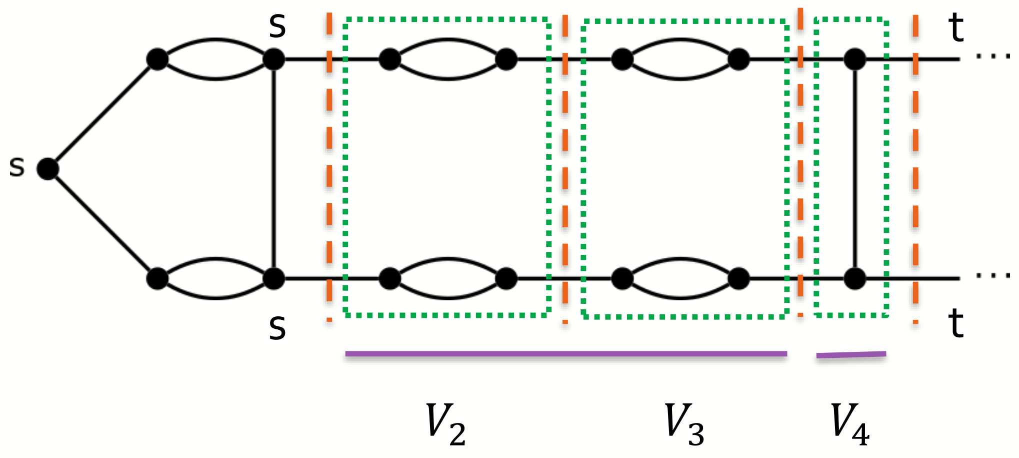
# Sequences of min-cuts

- Sequence of closest disjoint min-cuts $C_1, C_2, \ldots$
- **Blocks** $V_i$**:** Sets of vertices between $C_i$'s
- **Bundles:** Either
  1. Single connected block, or
  2. Sequence of blocks, just short of inducing a connected subgraph



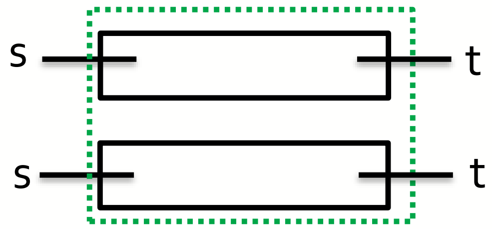
# Flow augmentation: Recursion

- **Claim:** At most $O(k)$ bundles are affected by $Z$
  - For any other bundle W, all vertices of N[W] are in one component
- A maximal sequence of affected bundles defines a **new flow augmentation instance** (we "zoom in" on part of the graph)
- **Recursion state** $(\lambda, k)$: Progress = (increase $\lambda$) or (decrease $k$)

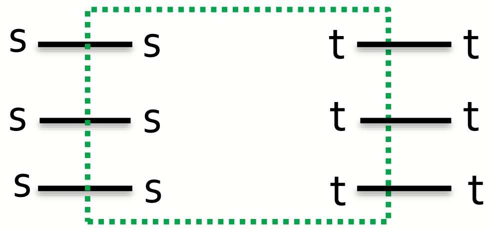


# Single affected bundle

- **Disconnected** bundle:
  - Recurse independently into each component
  - Each call decreases $k$

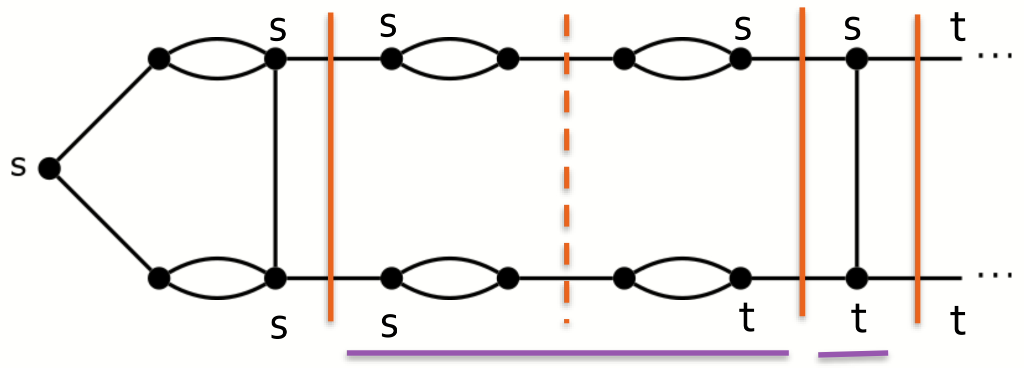



- **Connected** bundle (single block):
  1. **Surrounding edge** cut – decreases $k$
  2. **No surrounding edge cut** – increases $\lambda$



# Multiple affected bundles

- Select one cut $C_i$ between bundles
- Guess assignment (s/t) for all vertices ($O(\lambda)$ many)
  - After initial cut
  - Surrounding the cut $C_i$
  - Before the final cut




- Recurse into both halves
- Progress:
  - Because all bundles were affected, cut budget $k$ must be split two ways

# Summary

- **Flow Augmentation** – solve hard cut problems by reducing them to **min-cut/max-flow** instances
- Applications shown:
  - Weighted cut problems
  - Coupled Min-Cut / **Min SAT($\Gamma$)** trichotomy
- Is there **directed flow augmentation?**