Congestion-Free Network Updates: Algorithms and Complexity

Mitja Daniel Krebs

June 12, 2023

Contents

Table 1: Table of notations

$\leq_G$	The reachability relation for directed graph G
$b(v, P)$	The P-block containing vertex v
$P(b)$	The flow pair P such that block b is a P-block
$\mathcal{S}(b)$	The least vertex in block b w.r.t. reachability relation $\leq_{P(b)^o \cup P(b)^u}$
$B^P(G)$	The set of P-blocks
$B(G)$	The set of blocks
$\mathcal{B}(b)$	The round in which block b is updated
$\mathcal{B}(v, P)$	The round in which block $b(v, P)$ is updated
B_i	The set of blocks updated before or in the i-th round
$S(\pi)$	The set of which π is a permutation
$\|\pi\|$	The number of elements of permutation π
π_i	The i-th element of permutation π
$\pi(x)$	The position of element x in permutation π

Part I

Preliminaries

Chapter 1

Blocks

Notation 1. For a directed graph G, $\leq_G = E(G)^*$ denotes the reachability relation of G. That is, for every two vertices $u, v \in V(G)$, $u \leq_G v$ iff there is a path in G from u to v.

Notice that for every directed graph G,

1. reachability relation $\leq_G$ is a partial order,
2. if G is a DAG, then $\leq_G$ is antisymmetric, and
3. if G is a path graph, then $\leq_G$ is a total order.

Definition 2. Let $G = (V, E, \mathcal{P}, s, t, c)$ be an update flow network and $P \in \mathcal{P}$ be a flow pair. Let $v_1, \ldots, v_\ell$ be the set $V(P^o \cap P^u)$ ordered w.r.t. $\leq_{P^o \cup P^u}$. For every $i \in [\ell - 1]$, we define the i-th *P-block* as $b_i^P = \{v \mid v_i \leq_{P^o \cup P^u} v \leq_{P^o \cup P^u} v_{i+1}\}$.

Remark 3. There are multiple issues with this definition (see `../README.org`).

Notation 4. For a flow pair P and a vertex $v \in V(P)$, $b(v, P)$ denotes the P-block containing v.

Notation 5. For a block b, $P(b)$ denotes the flow pair P such that b is a P-block.

Notation 6. For a block b, $\mathcal{S}(b)$ denotes the *start* of b, that is, the least vertex in b w.r.t. $\leq_{P(b)^o \cup P(b)^u}$.

Notation 7. For an update flow network G and a flow pair P, $B^P(G)$ denotes the set of P-blocks.

Notation 8. For an update flow network G, $B(G) = \bigcup_{P \in \mathcal{P}} B^P(G)$ denotes the set of blocks.

Chapter 2

Block Sequences

Definition 9. A *block sequence* $\mathcal{B} = (\mathscr{B}_1, \ldots, \mathscr{B}_\ell)$ is an ordered partition of the set of blocks.

Remark 10. We may ignore all blocks containing less than three vertices.

☐ Flesh out and argue why.

Notation 11. For a block sequence $\mathcal{B} = (\mathscr{B}_1, \ldots, \mathscr{B}_\ell)$ and a block b, $\mathcal{B}(b)$ denotes the index $i \in [\ell]$ such that b is contained in $\mathscr{B}_i$.

Notation 12. For a block sequence $\mathcal{B} = (\mathscr{B}_1, \ldots, \mathscr{B}_\ell)$, a flow pair P, and a vertex $v \in V(P)$, $\mathcal{B}(v, P) = \mathcal{B}(b(v, P))$ denotes the index $i \in [\ell]$ such that block $b(v, P)$ is contained in $\mathscr{B}_i$.

Notation 13. For a block sequence $\mathcal{B} = (\mathscr{B}_1, \ldots, \mathscr{B}_\ell)$ and an index $i \in [\ell]$, $B_i = \bigcup_{j \leq i} \mathscr{B}_j$ denotes the set of blocks updated before or in the i-th round.

Definition 14. Let $\mathcal{B} = (\mathscr{B}_1, \ldots, \mathscr{B}_\ell)$ be a block sequence. For a flow pair P, an edge $(u, v) \in E(P^o \cup P^u)$, and an index $i \in [\ell]$, the *activation label* $\alpha_P((u, v), B_i)$ is defined as follows:

$$\alpha_P((u,v), B_i) = \begin{cases} \text{active} & \text{if } (u,v) \in E(P^o) \text{ and } b(u,P) \notin B_i \\ \text{active} & \text{if } (u,v) \in E(P^u) \text{ and } b(u,P) \in B_i \\ \text{inactive} & \text{otherwise.} \end{cases}$$

Lemma 15. *Let $\mathcal{B} = (\mathscr{B}_1, \ldots, \mathscr{B}_\ell)$ be a block sequence, P be a flow pair, $(u, v) \in E(P^o \cup P^u)$, and $i \in [\ell]$. Then:*

1. *If $(u, v) \in E(P^o \setminus P^u)$, then*

$$\alpha_P((u,v), B_i) = \begin{cases} \text{active} & i < \mathcal{B}(u,P) \\ \text{inactive} & i \geq \mathcal{B}(u,P). \end{cases}$$

2. *If* $(u,v) \in E(P^o \cap P^u)$*, then* $\alpha_P((u,v)B_i) = \text{active}$.
3. *If* $(u,v) \in E(P^u \setminus P^o)$*, then*

$$\alpha_P((u,v),B_i) = \begin{cases} \text{active} & i \geq \mathcal{B}(u,P) \\ \text{inactive} & i < \mathcal{B}(u,P). \end{cases}$$

Notation 16. For a flow pair P and a P-block b, $U(b) = \{(v,P) \mid v \in b\}$ denotes the set of updates induced by b. Moreover, for a set B of blocks, $U(B) = \bigcup_{b \in B} U(b)$ denotes the set of updates induced by B.

The following lemma shows that for every block sequence $\mathcal{B} = (\mathscr{B}_1, \ldots, \mathscr{B}_\ell)$, every flow pair P, every edge $e \in E(P^o \cup P^u)$, and every $i \in [\ell]$, $\alpha_P(e,B_i) =$ active iff e is on the transient (s,t)-path for P after updating all blocks in B_i.

Lemma 17. *Let* $\mathcal{B} = (\mathscr{B}_1, \ldots, \mathscr{B}_\ell)$ *be a block sequence,* P *be a flow pair,* $e \in E(P^o \cup P^u)$*, and* $i \in [\ell]$*. Then* $\alpha_P(e,B_i) = \text{active}$ *iff* $e \in E(T_{P,U(B_i)})$.

Definition 18. A block sequence $\mathcal{B} = (\mathscr{B}_1, \ldots, \mathscr{B}_\ell)$ is *feasible* if for every edge e and every index $i \in [\ell]$,

$$c(e) \geq \sum_{P \in \mathcal{P} : \alpha_P(e,B_{i-1}) = \text{active or } \alpha_P(e,B_i) = \text{active}} d_P, \tag{2.1}$$

where we define $\mathscr{B}_0$ to be the empty set.

Remark 19. Let G be an update flow network with unit demand, that is, $d_P = 1$ for every flow pair P, and let $\mathcal{B} = (\mathscr{B}_1, \ldots, \mathscr{B}_\ell)$ be a block sequence. Then, for every edge e and every index $i \in [\ell]$, capacity constraint 2.1 simplifies to:

$$\begin{aligned} c(e) &\geq \sum_{P \in \mathcal{P} : \alpha_P(e,B_{i-1}) = \text{active or } \alpha_P(e,B_i) = \text{active}} d_P \\ &= \sum_{P \in \mathcal{P} : \alpha_P(e,B_{i-1}) = \text{active or } \alpha_P(e,B_i) = \text{active}} 1 \\ &= |\{P \in \mathcal{P} \mid \alpha_P(e,B_{i-1}) = \text{active or } \alpha_P(e,B_i) = \text{active}\}|. \end{aligned}$$

Lemma 20. *Let* G *be a not necessarily feasible update flow network and* $\mathcal{B} = (\mathscr{B}_1, \ldots, \mathscr{B}_\ell)$ *be a block sequence. Then:*

1. *The old flow network is feasible if capacity constraint 2.1 is satisfied for every edge and* $i = 1$.
2. *The updated flow network is feasible if capacity constraint 2.1 is satisfied for every edge and* $i = \ell$.

Proof. Let G be a not necessarily feasible update flow network and $\mathcal{B} = (\mathscr{B}_1, \ldots, \mathscr{B}_\ell)$ be a block sequence. Moreover, let e be an edge.

1. Suppose capacity constraint 2.1 is satisfied for e and $i = 1$. Then, since $\mathscr{B}_0 = \emptyset$, and by definitions of B_i and α_P:

$$\begin{aligned} c(e) &\geq \sum_{P \in \mathcal{P} : \alpha_P(e, B_0) = \text{active or } \alpha_P(e, B_1) = \text{active}} d_P \\ &\geq \sum_{P \in \mathcal{P} : \alpha_P(e, B_0) = \text{active}} d_P \\ &= \sum_{P \in \mathcal{P} : e \in E(P^o)} d_P. \end{aligned}$$

2. Suppose capacity constraint 2.1 is satisfied for e and $i = \ell$. Then, since $\mathcal{B}$ partitions the set of blocks, and by definitions of B_i and α_P:

$$\begin{aligned} c(e) &\geq \sum_{P \in \mathcal{P} : \alpha_P(e, B_{\ell-1}) = \text{active or } \alpha_P(e, B_\ell) = \text{active}} d_P \\ &\geq \sum_{P \in \mathcal{P} : \alpha_P(e, B_\ell) = \text{active}} d_P \\ &= \sum_{P \in \mathcal{P} : e \in E(P^u)} d_P. \end{aligned}$$

□

Corollary 21. *There is a feasible block sequence iff there is a feasible update sequence.*

Chapter 3

Block Permutations

Notation 22. For a set S and a permutation $\pi = (x_1, \dots, x_\ell)$ of S,

1. $S(\pi)$ denotes the set S of which π is a permutation;
2. $|\pi|$ denotes the number ℓ of elements of π;
3. for an index $i \in [\ell]$, π_i denotes the i-th element x_i of π; and
4. for an element $x \in S$, $\pi(x)$ denotes the index i such that $\pi_i = x$.

Notation 23. For two permutations π_1, π_2, $\operatorname{core}(\pi_1, \pi_2) = S(\pi_1) \cap S(\pi_2)$ denotes the *core* of π_1 and π_2.

Definition 24. Let π be a permutation and π' be a subsequence of π. Then:

1. π is an *extension* of π' to $S(\pi) \supseteq S(\pi')$; and
2. π' is the *restriction* of π to $S(\pi') \subseteq S(\pi)$.

☐ Should we define subsequence?

☐ Remark something about empty permutations.

Definition 25. Two permutations π_1, π_2 are *consistent* if the restrictions of π_1 and π_2 to $\operatorname{core}(\pi_1, \pi_2)$ are equal.

Lemma 26. *Let π_1, π_2, π_3 be three permutations such that $S(\pi_1) \subseteq S(\pi_2) \subseteq S(\pi_3)$. Then:*

1. *If π_3 is an extension of π_2 and π_2 is an extension of π_1, then π_3 is an extension of π_2.*
2. *If π_3 is an extension of both π_1 and π_2, then π_2 is an extension of π_1.*

Proof. Let π_1, π_2, π_3 be three permutations such that $S(\pi_1) \subseteq S(\pi_2) \subseteq S(\pi_3)$.

1.

2. □

Definition 27. Let π_1, π_2 be two consistent permutations. A permutation π is a *union* of π_1 and π_2 if π is an extension of both π_1 and π_2 to $S(\pi_1) \cup S(\pi_2)$.

Definition 28. Let $X \subseteq E(G)$ be a set of edges. A permutation $\pi = (b_1, \ldots, b_\ell)$ of blocks is *congestion free* w.r.t. X if for every edge $e \in X$ and every index $i \in [\ell]$,

$$c(e) \geq \sum_{P \in \mathcal{P} : \alpha_P(e, B_i) = \text{active}} d_P, \tag{3.1}$$

where $B_i = \bigcup_{j \leq i} \{b_i\}$.

□ Remark why we may overload notation B_i.

Lemma 29. *Let $X \subseteq E(G)$ be a set of edges and $\pi = (b_1, \ldots, b_\ell)$ be a permutation of blocks. Then π is congestion free w.r.t. X iff the block sequence $\mathcal{B} = (\{b_1\}, \ldots, \{b_\ell\})$ induced by π is feasible w.r.t. X.*

□ The block sequence induced by π is not defined unless $S(\pi) = B(G)$.

□ Feasible w.r.t. X is not defined unless $X = E(G)$.

Proof. Let $X \subseteq E(G)$ be a set of edges, $\pi = (b_1, \ldots, b_\ell)$ be a permutation, and $e \in X$ be an edge.

Only-if part. Let $i \in [\ell]$. If capacity constraint 2.1 is satisfied for the block sequence $\mathcal{B} = (\{b_1\}, \ldots, \{b_\ell\})$ induced by π, e, and i, then capacity constraint 3.1 is satisfied for π, e, and i:

$$\sum_{P \in \mathcal{P} : \alpha_P(e, B_i) = \text{active}} d_P \leq \sum_{P \in \mathcal{P} : \alpha_P(e, B_{i-1}) = \text{active or } \alpha_P(e, B_i) = \text{active}} d_P \leq c(e)$$

If part. We show the contrapositive. Suppose capacity contraint 2.1 is violated for some $i \in [\ell]$. We show that capacity constraint 3.1 is violated for i or $i - 1$. Notice that if $i = 1$, then the latter contradicts the feasibility of update flow network G.

Since block b_i is the only block updated in round i, we have that for every block $b \neq b_i$, $b \in B_i$ iff $b \in B_{i-1}$. Hence for every flow pair $P \in \mathcal{P} \setminus \{P(b_i)\}$, we have $\alpha_P(e, B_i) = \text{active}$ iff $\alpha_P(e, B_{i-1}) = \text{active}$. For flow pair $P(b_i)$, we consider the cases $\alpha_{P(b_i)}(e, B_i) = \text{active}$ and $\alpha_{P(b_i)}(e, B_{i-1}) = \text{active}$ separately. (Note that if neither $\alpha_{P(b_i)}(e, B_i) = \text{active}$ nor $\alpha_{P(b_i)}(e, B_{i-1}) = \text{active}$, then demand $d_{P(b_i)}$ contributes to neither sum.)

If $\alpha_{P(b_i)}(e, B_i) = \text{active}$, then capacity constraint 3.1 is violated for i:

$$\begin{aligned}
&\sum_{P\in\mathcal{P}:\alpha_P(e,B_i)=\text{active}} d_P = \\
&\sum_{P\in\mathcal{P}\setminus\{P(b_i)\}:\alpha_P(e,B_i)=\text{active}} d_P + \sum_{P\in\{P(b_i)\}:\alpha_P(e,B_i)=\text{active}} d_P = \\
&\sum_{P\in\mathcal{P}\setminus\{P(b_i)\}:\alpha_P(e,B_{i-1})=\text{active or }\alpha_P(e,B_i)=\text{active}} d_P \\
&+ \sum_{P\in\{P(b_i)\}:\alpha_P(e,B_{i-1})=\text{active or }\alpha_P(e,B_i)=\text{active}} d_P = \\
&\sum_{P\in\mathcal{P}:\alpha_P(e,B_{i-1})=\text{active or }\alpha_P(e,B_i)=\text{active}} d_P > c(e).
\end{aligned}$$

If $\alpha_{P(b_i)}(e, B_{i-1}) = \text{active}$, then capacity constraint 3.1 is violated for $i-1$:

$$\begin{aligned}
&\sum_{P\in\mathcal{P}:\alpha_P(e,B_{i-1})=\text{active}} d_P = \\
&\sum_{P\in\mathcal{P}\setminus\{P(b_i)\}:\alpha_P(e,B_{i-1})=\text{active}} d_P + \sum_{P\in\{P(b_i)\}:\alpha_P(e,B_{i-1})=\text{active}} d_P = \\
&\sum_{P\in\mathcal{P}\setminus\{P(b_i)\}:\alpha_P(e,B_{i-1})=\text{active or }\alpha_P(e,B_i)=\text{active}} d_P \\
&+ \sum_{P\in\{P(b_i)\}:\alpha_P(e,B_{i-1})=\text{active or }\alpha_P(e,B_i)=\text{active}} d_P = \\
&\sum_{P\in\mathcal{P}:\alpha_P(e,B_{i-1})=\text{active or }\alpha_P(e,B_i)=\text{active}} d_P > c(e).
\end{aligned}$$

□

Definition 30. Let $X \subseteq V(G)$ be a set of vertices. A permutation π is:

1. an X-*permutation* if it is a permutation of the blocks $\bigcup_{P\in\mathcal{P},v\in X}\{b(v,P)\}$ induced by X; and
2. X-*feasible* if it is an X-permutation and congestion free w.r.t. $E(X)$.

□ I think we need to be careful here whether $E(X)$ comprises directed or undirected edges.

Definition 31. Let $X \subseteq V(G)$ be a set of vertices and π, π' be two permutations. Then:

1. π is an X-*extension* of π' if π is an extension of π' to X; and
2. π' and the X-*restriction* of π if π' is the extension of π to X.

Lemma 32. *Let $X, Y, Z \subseteq V(G)$ be three sets of vertices such that $X \subseteq Y \subseteq Z$. Moreover, let π_X be an X-permutation, π_Y be a Y-permutation, and π_Z be a Z-permutation. Then:*

1. *If π_Z is an extension of π_Y and π_Y is an extension of π_X, then π_Z is an extension of π_X.*
2. *If π_Z is an extension of both π_X and π_Y, then π_Y is an extension of π_X.*

Proof. The lemma follows immediately from Lemma 26 and the fact that if $X \subseteq Y$, then $\bigcup_{P\in\mathcal{P},v\in X}\{b(v,P)\} \subseteq \bigcup_{P\in\mathcal{P},v\in Y}\{b(v,P)\}$. □

Part II

NP-Hardness for $k = 3$

The goal of this section is to prove the following theorem.

Theorem 33. *The k-network flow update problem is **NP**-hard for $k = 3$.*

We will prove this theorem in two steps. First, we will prove the following theorem.

Theorem 34. *The k-network flow update problem, where every edge is used by at most three flow pairs, is **NP**-hard for $k = 10$.*

Then, we will (repeatedly) apply the following lemma to the flow update network we will have constructed in the proof of Theorem 34 to reduce the number of flow pairs from 10 to 3.

Lemma 35 (Merging Lemma)**.** *Let G be an update flow network with $k \geq 2$ flow pairs, and let F, F' be two flow pairs such that*

1. *$d_F = d_{F'}$,*
2. *F and F' have no common vertices other than s, t, that is, $V(F^o \cup F^u) \cap V(F'^o \cup F'^u) = \{s, t\}$, and*
3. *there are vertices $v_F, v_{F'}$ such that*
   - *(a) there is no edge from v_F to $v_{F'}$, that is, $(v_F, v_{F'}) \notin E$,*
   - *(b) (v_F, t) ($(s, v_{F'})$) is the last (first) edge on both F^o and F^u (F'^o and F'^u), that is, $(v_F, t) \in E(F^o \cap F^u)$ ($(s, v_{F'}) \in E(F'^o \cap F'^u)$), and*
   - *(c) the capacity constraint for (v_F, t) ($(s, v_{F'})$) is trivially satisfied, that is,*
   $$c(e) \geq \sum_{P \in \mathcal{P} : e \in E(P^o \cup P^u)} d_P$$
   for $e = (v_F, t)$ ($e = (s, v_{F'})$).

Then there is an update flow network $\tilde{G}$ with $k - 1$ flow pairs such that ($|\tilde{G}| = O(|G|)$ and) there is a feasible block sequence $\mathcal{B} = (\mathscr{B}_1, \ldots, \mathscr{B}_\ell)$ for G iff there is a feasible block sequence $\tilde{\mathcal{B}} = (\tilde{\mathscr{B}}_1, \ldots, \tilde{\mathscr{B}}_\ell)$ for $\tilde{G}$.

Remark 36. I'm confident we don't need property 3, but it significantly simplifies the proof.

Chapter 4

NP-Hardness for the Special Case

The proof of Theorem 34 is via reduction from 4-SAT and is based on the **NP**-hardness proof for $k = 6$ in (Amiri, Saeed A. and Dudycz, Szymon and Parham, Mahmoud and Schmid, Stefan and Wiederrecht, Sebastian, 2019).

Let C be a 4CNF formula with n variables $x_1, \ldots, x_n$ and m clauses $C_1, \ldots, C_m$. W.l.o.g. every variable occurs both positively and negatively (otherwise, if a variable x_j occurs only positively (negatively), we can assign 1 (0) to x_j and remove all clauses containing literal x_j ($\bar{x}_j$)). We construct the corresponding update flow network G as follows.

4.1 The Reduction

We will need to map a solution for formula C, that is, a satisfying assignment, to a solution for update flow network G, that is, a feasible block sequence, and vice versa. We will do this by mapping the choice of assigning either 1 or 0 to variable x_j to the choice of updating a corresponding block, say b^j, after a dedicated reference block b or not, and vice versa. For this purpose, we construct a *variable gadget* X^j for every variable x_j such that, in particular, block b^j is completely contained in X^j, whereas reference block b spans all variable gadgets. Moreover, to guarantee that an assignment obtained from a feasible block sequence as indicated above indeed satisfies every clause, we additionally construct a *clause gadget* for every clause. We first describe the variable gadgets.

☐ Mention that all flow pairs have unit demand.

Variable gadgets. For every variable x_j, we construct the corresponding variable gadget X^j as follows (see Figure 4.1). As indicated above, the idea is that X^j contains a block b^j such that:

(V1) Variable x_j is assigned 1 iff block b^j is updated after reference block b.

Therefore, we introduce a *variable vertex* x^j which is used by three flow pairs $X, \bar{X}, B$ such that $b^j = b(x^j, \bar{X})$ and $b = b(x^j, B)$. Intuitively, updating flow pair X ($\bar{X}$) before B for x^j corresponds to assigning 1 (0) to variable x_j. More precisely, the idea is to guarantee the following:

(V2) If block $b(x^j, X)$ is updated before $b(x^j, B)$, then variable x_j is assigned 1.

(V3) If $b(x^j, \bar{X})$ is updated before $b(x^j, B)$, then x_j is assigned 0.

(V4) Not both $b(x^j, X)$ and $b(x^j, \bar{X})$ can be updated before $b(x^j, B)$.

Notice that (V1) and (V4) imply both (V2) and (V3). To guarantee (V4), we introduce a vertex y^j and add an edge (x^j, y^j) with capacity 2 to flows $X^u, \bar{X}^u, B^o$.

Moreover, we introduce auxiliary vertices $x_0^j, y_0^j, x_1^j, y_1^j$ and add edge (x_0^j, y_0^j) with capacity 1 to flow $\bar{X}^o$ and (x_1^j, y_1^j) with capacity 1 to X^o. These will be used to connect variable with clause gadgets. More precisely, x_0^j (x_1^j) will be connected to clause gadgets corresponding to clauses in which variable x_j occurs negatively (positively).

As mentioned above, block b^j should be completely contained in variable gadget X^j. Therefore, we add edges $(x^j, x_0^j), (y_0^j, y^j)$ with capacity 10 to flow $\bar{X}^o$ and $(x^j, x_1^j), (y_1^j, y^j)$ with capacity 10 to X^o.

☐ Either use capacity ∞ or argue why we use capacity 10.

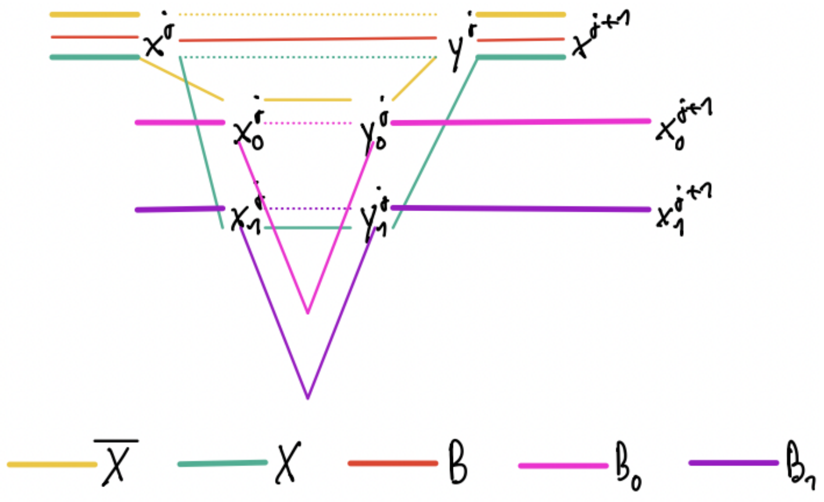

Figure 4.1: Variable gadget X^j

Clause gadgets. Let $C_i = (l_{i_1} \vee l_{i_2} \vee l_{i_3} \vee l_{i_4})$ be a clause. We construct the corresponding clause gadget C^i as follows (see Figure 4.5). The idea is to model the syntax tree for C_i depicted in Figure 4.2.

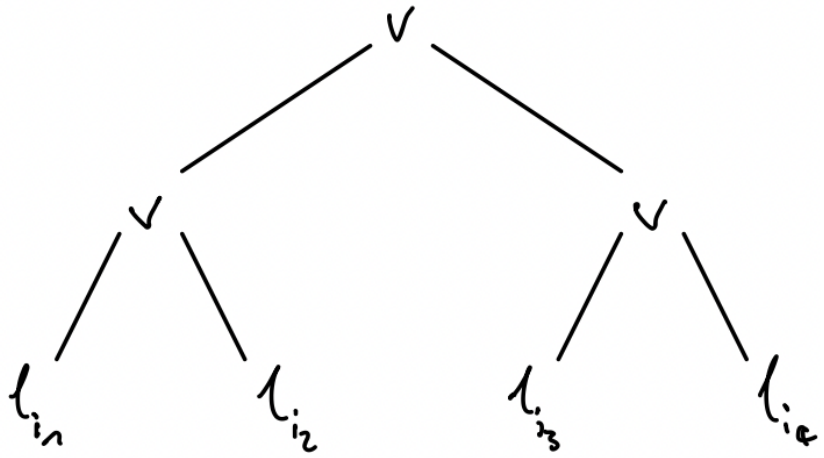

Figure 4.2: A syntax tree for clause $(l_{i_1} \vee l_{i_2} \vee l_{i_3} \vee l_{i_4})$

For the root operator node, we introduce a *clause vertex* u^i which is used by three flow pairs L, R, B such that block $b(u^i, B)$ equals the reference block b. The idea is to guarantee the following:

(C1) Clause C_i is satisfied iff block $b(u^i, L)$ is updated before b or $b(u^i, R)$ is updated before b.

Intuitively, if $b(u^i, L)$ ($b(u^i, R)$) is updated before b, then the **L**eft half $(l_{i_1} \vee l_{i_2})$ (**R**ight half $(l_{i_3} \vee l_{i_4})$) of C_i is satisfied. Similarly, for the intermediate operator nodes of the syntax tree, we introduce clause vertices u^i_L, u^i_R, where u^i_L corresponds to the left half $(l_{i_1} \vee l_{i_2})$ of C_i and u^i_R corresponds to the right half $(l_{i_3} \vee l_{i_4})$. Both vertices are used by flow pairs $\tilde{L}, \tilde{R}, \tilde{B}$ such that:

(C2) Suppose block $b(u^i, L)$ is updated before b. Then clause $(l_{i_1} \vee l_{i_2})$ is satisfied if $b(u^i_L, \tilde{L})$ is updated before $b(u^i_L, \tilde{B})$ or $b(u^i_L, \tilde{R})$ is updated before $b(u^i_L, \tilde{B})$.

(C3) Suppose block $b(u^i, R)$ is updated before b. Then clause $(l_{i_3} \vee l_{i_4})$ is satisfied if $b(u^i_R, \tilde{L})$ is updated before $b(u^i_R, \tilde{B})$ or $b(u^i_R, \tilde{R})$ is updated before $b(u^i_R, \tilde{B})$.

Again, intuitively, if $b(u^i_L, \tilde{L})$ ($b(u^i_L, \tilde{R})$) is updated before $b(u^i_L, \tilde{B})$, then the left half l_{i_1} (right half l_{i_2}) of $(l_{i_1} \vee l_{i_2})$ is satisfied, and analogously for u^i_R.

Moreover, for the operand nodes of the syntax tree, we introduce *literal vertices* $u^i_1, u^i_2, u^i_3, u^i_4$. Finally, for every branch from a parent node to its left

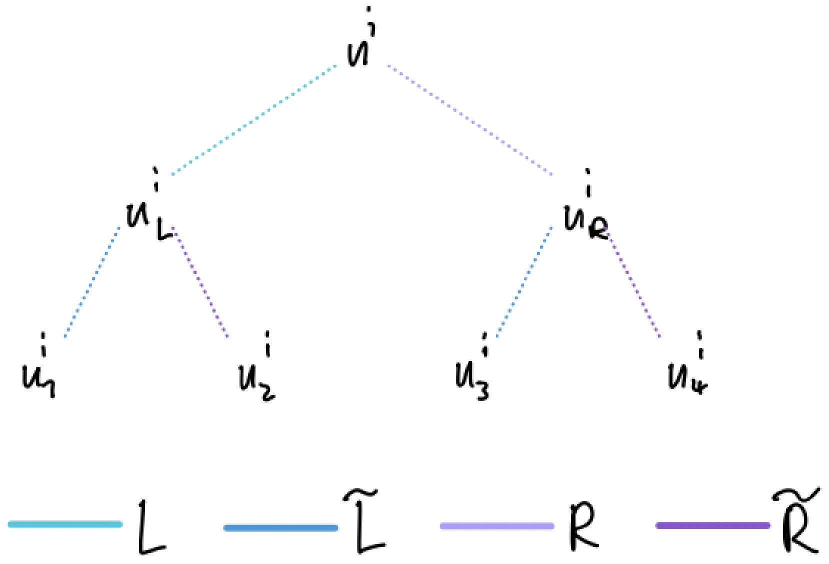

Figure 4.3: Clause gadget C^i

(right) child node, we add an edge to either L^u (R^u) (if the parent node is u^i) or $\tilde{L}^u$ ($\tilde{R}^u$) (if the parent node is u^i_L or u^i_R). See Figure 4.3 for an illustration.

To guarantee (C1), we introduce a vertex v^i and add an edge (u^i, v^i) with capacity 2 to flows L^o, R^o, B^u. Similarly, to guarantee (C2) and (C3), we introduce vertices v^i_L, v^i_R and add edges $(u^i_L, v^i_L), (u^i_R, v^i_R)$ with capacity 2 to flows $\tilde{L}^o, \tilde{R}^o, \tilde{B}^u$. See Figure 4.4 for an illustration.

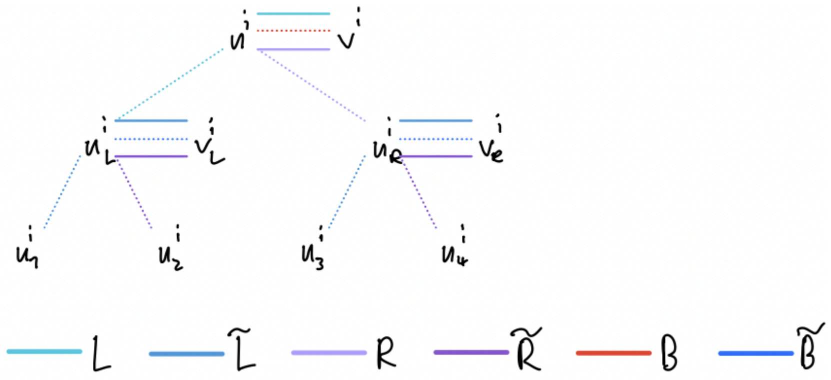

Figure 4.4: Clause gadget C^i

Moreover, we introduce vertices $v^i_1, v^i_2, v^i_3, v^i_4$ and add edges $(u^i_1, v^i_1), (u^i_3, v^i_3)$ with capacity 1 to flow $\tilde{L}^u$ and $(u^i_2, v^i_2), (u^i_4, v^i_4)$ with capacity 1 to $\tilde{R}^u$. These will be used to connect variable with clause gadgets.

Finally, we introduce auxiliary vertices $\tilde{u}^i_L, \tilde{v}^i_L, \tilde{u}^i_R, \tilde{v}^i_R$ and add edge $(\tilde{u}^i_L, \tilde{v}^i_L)$ with capacity 1 to flows $L^u, \tilde{B}^o$ and $(\tilde{u}^i_R, \tilde{v}^i_R)$ with capacity 1 to $R^u, \tilde{B}^o$. We remark that these vertices, as well as flow pair $\tilde{B}$, are not necessary for this proof. Instead, we could directly connect clause vertices u^i, u^i_L via flow pair L and u^i, u^i_R via R. They are necessary, however, for the proof of Theorem 33.

Since we want to guarantee (C1), (C2), and (C3) for each clause independently, blocks $b(u^i, L), b(u^i, R), b(u^i_L, \tilde{L}), b(u^i_L, \tilde{R}), b(u^i_R, \tilde{L}), b(u^i_R, \tilde{R})$ should be completely contained in clause gadget C^i. Moreover, since we want to guarantee (C2) and (C3) independently, blocks $b(u^i_L, \tilde{L}), b(u^i_R, \tilde{L})$, $b(u^i_L, \tilde{R}), b(u^i_R, \tilde{R})$, and $b(u^i_L, \tilde{B}), b(u^i_R, \tilde{B})$ should be distinct. Therefore, we add the following edges, each with capacity 10:

- $(u^i, \tilde{u}^i_L), (\tilde{v}^i_L, v^i)$ to L^u
- $(u^i, \tilde{u}^i_R), (\tilde{v}^i_R, v^i)$ to R^u
- $(u^i_L, u^i_1), (v^i_1, v^i_L), (u^i_R, u^i_3), (v^i_3, v^i_R)$ to $\tilde{L}^u$
- $(u^i_L, u^i_2), (v^i_2, v^i_L), (u^i_R, u^i_4), (v^i_4, v^i_R)$ to $\tilde{R}^u$
- $(\tilde{u}^i_L, u^i_L), (v^i_L, \tilde{v}^i_L), (\tilde{u}^i_R, u^i_R), (v^i_R, \tilde{v}^i_R)$ to $\tilde{B}^u$

Finally, to obtain (s,t)-paths in the end, we add the following edges, each with capacity 10:

- (v^i_L, u^i_R) to $\tilde{L}^o, \tilde{L}^u, \tilde{R}^o, \tilde{R}^u$
- $(\tilde{v}^i_L, \tilde{u}^i_R)$ to $\tilde{B}^o, \tilde{B}^u$

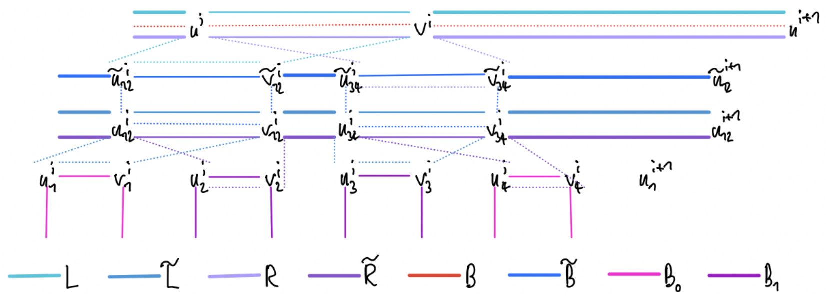

Figure 4.5: Clause gadget C^i

☐ Merge clause gadget figures.

Connecting variable with clause gadgets. For every $j \in [n]$ and every $i \in [m]$, we connect variable gadget X^j to clause gadget C^i if variable x_j occurs in clause C_i. More precisely, we introduce two flow pairs B_0, B_1 such that B_0 (B_1) connects vertex x_0^j (x_1^j) to all literal vertices corresponding to literal $\bar{x}_j$ (x_j). More formally, for every $j \in [n]$, let $P_j = \{p_1^j, \ldots, p_{\ell_j}^j\}$ denote the set of indices of the clauses containing literal x_j and $\bar{P}_j = \{\bar{p}_1^j, \ldots, \bar{p}_{\ell'_j}^j\}$ denote the set of indices of the clauses containing literal $\bar{x}_j$. Moreover, for every $j \in [n]$ and every $i \in [m]$, let $\pi(i,j)$ denote the position of literal x_j in clause C_i and $\bar{\pi}(i,j)$ denote the position of literal $\bar{x}_j$ in C_i. For every $j \in [n]$, we add the following edges:

- $(x_0^j, u^{\bar{p}_1^j}_{\bar{\pi}(\bar{p}_1^j,j)})$ with capacity 10, $(u^{\bar{p}_\ell^j}_{\bar{\pi}(\bar{p}_\ell^j,j)}, v^{\bar{p}_\ell^j}_{\bar{\pi}(\bar{p}_\ell^j,j)})$ for every $\ell \in [\ell'_j]$, $(v^{\bar{p}_\ell^j}_{\bar{\pi}(\bar{p}_\ell^j,j)}, u^{\bar{p}_{\ell+1}^j}_{\bar{\pi}(\bar{p}_{\ell+1}^j,j)})$ with capacity 10 for every $\ell \in [\ell'_j - 1]$, and $(v^{\bar{p}_{\ell'_j}^j}_{\bar{\pi}(\bar{p}_{\ell'_j}^j,j)}, y_0^j)$ with capacity 10 to B_0^o

- $(x_1^j, u^{p_1^j}_{\pi(p_1^j,j)})$ with capacity 10, $(u^{p_\ell^j}_{\pi(p_\ell^j,j)}, v^{p_\ell^j}_{\pi(p_\ell^j,j)})$ for every $\ell \in [\ell_j]$, $(v^{p_\ell^j}_{\pi(p_\ell^j,j)}, u^{p_{\ell+1}^j}_{\pi(p_{\ell+1}^j,j)})$ with capacity 10 for every $\ell \in [\ell_j - 1]$, and $(v^{p_{\ell_j}^j}_{\pi(p_{\ell_j}^j,j)}, y_1^j)$ with capacity 10 to B_1^o

Putting everything together. We introduce vertices s, t and create (s,t)-paths for all flows by adding the following edges:

- $(s, u^1), (v^m, t)$ to L^o, L^u, R^o, R^u
- (v^i, u^{i+1}) for every $i \in [m-1]$ to L^o, L^u, R^o, R^u, B^u
- (s, u_L^1), (v_R^i, u_L^{i+1}) for every $i \in [m-1]$, and (v_R^m, t) to $\tilde{L}^o, \tilde{L}^u, \tilde{R}^o, \tilde{R}^u$
- $(s, \tilde{u}_L^1)$, $(\tilde{v}_R^i, \tilde{u}_L^{i+1})$ for every $i \in [m-1]$, and $(\tilde{v}_R^m, t)$ to $\tilde{B}^o, \tilde{B}^u$
- $(s, x^1), (y^n, t)$ to $X^o, X^u, \bar{X}^o, \bar{X}^u, B^o, B^u$
- (y^j, x^{j+1}) for every $j \in [n-1]$ to $X^o, X^u, \bar{X}^o, \bar{X}^u, B^o$
- $(x^1, u^1), (v^m, y^n)$ to B^u
- (s, x_0^1), (y_0^j, x_0^{j+1}) for every $j \in [n-1]$, and (y_0^n, t) to B_0^o, B_0^u
- (s, x_1^1), (y_1^j, x_1^{j+1}) for every $j \in [n-1]$, and (y_1^n, t) to B_1^o, B_1^u

See Figure 4.6 for the complete update flow network and Table 4.1 for all (s,t)-flows.

Edge capacities are defined as follows.

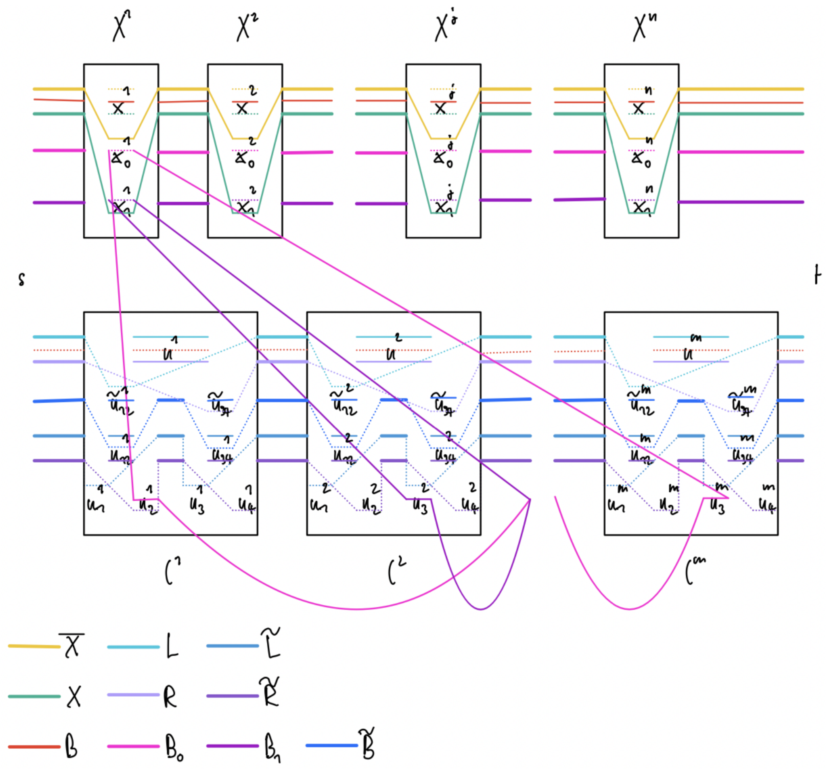

Figure 4.6: The update flow network

Table 4.1: All (s,t)-flows

Flow	(s,t)-path
$\bar{X}^o$	$s, x^1, x_0^1, y_0^1, y^1, x^2, \dots, y^n, t$
$\bar{X}^u$	$s, x^1, y^1, x^2, \dots, y^n, t$
L^o	$s, u^1, v^1, u^2, \dots, v^m, t$
L^u	$s, u^1, \tilde{u}_L^1, \tilde{v}_L^1, v^1, u^2, \dots, v^m, t$
$\tilde{L}^o$	$s, u_L^1, v_L^1, u_R^1, v_R^1, u_L^2, \dots, v_R^m, t$
$\tilde{L}^u$	$s, u_L^1, u_1^1, v_1^1, v_L^1, u_R^1, u_3^1, v_3^1, v_R^1, u_L^2, \dots, v_R^m, t$
X^o	$s, x^1, x_1^1, y_1^1, y^1, x^2, \dots, y^n, t$
X^u	$s, x^1, y^1, x^2, \dots, y^n, t$
R^o	$s, u^1, v^1, u^2, \dots, v^m, t$
R^u	$s, u^1, \tilde{u}_R^1, \tilde{v}_R^1, v^1, u^2, \dots, v^m, t$
$\tilde{R}^o$	$s, u_L^1, v_L^1, u_R^1, v_R^1, u_L^2, \dots, v_R^m, t$
$\tilde{R}^u$	$s, u_L^1, u_2^1, v_2^1, v_L^1, u_R^1, u_4^1, v_4^1, v_R^1, u_L^2, \dots, v_R^m, t$
B^o	$s, x^1, y^1, x^2, \dots, y^n, t$
B^u	$s, x^1, u^1, v^1, u^2, \dots, v^m, y^n, t$
$\tilde{B}^o$	$s, \tilde{u}_L^1, \tilde{v}_L^1, \tilde{u}_R^1, \tilde{v}_R^1, \tilde{u}_L^2, \dots, \tilde{v}_R^m, t$
$\tilde{B}^u$	$s, \tilde{u}_L^1, u_L^1, v_L^1, \tilde{v}_L^1, \tilde{u}_R^1, u_R^1, v_R^1, \tilde{v}_R^1, \tilde{u}_L^2, \dots, \tilde{v}_R^m, t$
B_0^o	$s, x_0^1, u_{\bar{\pi}(\bar{p}_1^1,1)}^{\bar{p}_1^1}, v_{\bar{\pi}(\bar{p}_1^1,1)}^{\bar{p}_1^1}, u_{\bar{\pi}(\bar{p}_2^1,1)}^{\bar{p}_2^1}, \dots, v_{\bar{\pi}(\bar{p}_{l'_1}^1,1)}^{\bar{p}_{l'_1}^1}, y_0^1, x_0^2, \dots, y_0^n, t$
B_0^u	$s, x_0^1, y_0^1, x_0^2, \dots, y_0^n, t$
B_1^o	$s, x_1^1, u_{\pi(p_1^1,1)}^{p_1^1}, v_{\pi(p_1^1,1)}^{p_1^1}, u_{\pi(p_2^1,1)}^{p_2^1}, \dots, v_{\pi(p_{l_1}^1,1)}^{p_{l_1}^1}, y_1^1, x_1^2, \dots, y_1^n, t$
B_1^u	$s, x_1^1, y_1^1, x_1^2, \dots, y_1^n, t$

- We set the capacity to 2 for edges $(u^i, v^i), (u_L^i, v_L^i), (u_R^i, v_R^i), (x^j, y^j)$ for every $i \in [m]$ and every $j \in [n]$.
- We set the capacity to 1 for edges $(u_1^i, v_1^i), (u_2^i, v_2^i), (u_3^i, v_3^i), (u_4^i, v_4^i), (\tilde{u}_L^i, \tilde{v}_L^i), (\tilde{u}_R^i, \tilde{v}_R^i), (x_0^j, y_0^j), (x_1^j, y_1^j)$ for every $i \in [m]$ and every $j \in [n]$.
- All remaining edge capacities are set to 10, that is, the number of flow pairs, which equals the sum of all demands.

We remark that vertices $\tilde{u}_L^i, \tilde{v}_L^i, \tilde{u}_R^i, \tilde{v}_R^i$ are not necessary for this proof. Instead, we could directly connect clause vertices u^i, u_L^i via flow pair L and u^i, u_R^i via R. Similarly, vertices $x_0^j, y_0^j, x_1^j, y_1^j$ as well as flow pairs B_0, B_1 are not necessary. We could instead directly connect variable vertex x^j to literal vertex, say u_1^i, via X ($\bar{X}$) if $l_{i_1} = x_j$ ($l_{i_1} = \bar{x}_j$). The vertices and flow pairs are necessary, however, for the proof of Theorem 33.

Let us quickly verify that G is a feasible update flow network.

To verify that every flow is indeed an (s,t)-path, see Table 4.1. Recall we assumed every variable x_j occurs both negatively and positively in formula C. Hence both $\bar{P}_j$ and P_j are non-empty. Thus both B_0^o and B_1^o form (s,t)-paths.

To verify that every flow pair forms a DAG, again consider Table 4.1.

Using Lemma 20, we will show that all capacity constraints are satisfied for both the old flow network and the updated flow network in the if part of the proof of Theorem 34.

4.2 The Proof

Before we prove Theorem 34, let us show that every feasible block sequence for the update flow network specified in the previous section satisfies the following properties.

Lemma 37. *Let $\mathcal{B}$ be a feasible block sequence for update flow network G. Then:*

1. *For every $i \in [m]$, $\mathcal{B}(u^i, L) < \mathcal{B}(x^1, B)$ or $\mathcal{B}(u^i, R) < \mathcal{B}(x^1, B)$.*
2. *For every $i \in [m]$,*
   (a) *$\mathcal{B}(\tilde{u}_L^i, \tilde{B}) < \mathcal{B}(u^i, L)$, and*
   (b) *$\mathcal{B}(\tilde{u}_R^i, \tilde{B}) < \mathcal{B}(u^i, R)$.*
3. *For every $i \in [m]$,*
   (a) *$\mathcal{B}(u_L^i, \tilde{L}) < \mathcal{B}(\tilde{u}_L^i, \tilde{B})$ or $\mathcal{B}(u_L^i, \tilde{R}) < \mathcal{B}(\tilde{u}_L^i, \tilde{B})$, and*
   (b) *$\mathcal{B}(u_R^i, \tilde{L}) < \mathcal{B}(\tilde{u}_R^i, \tilde{B})$ or $\mathcal{B}(u_R^i, \tilde{R}) < \mathcal{B}(\tilde{u}_R^i, \tilde{B})$.*
4. *For every $j \in [n]$, $\mathcal{B}(x^1, B) < \mathcal{B}(x^j, \bar{X})$ or $\mathcal{B}(x^1, B) < \mathcal{B}(x^j, X)$.*
5. *For every $i \in [m]$ and every $j \in [n]$,*

Table 4.2: All blocks grouped by flow pair

P	$V(P^o \cap P^u)$ ordered w.r.t. $\leq_{P^o \cup P^u}$	$B^P(G)$
$\bar{X}$	$s, x^1, y^1, x^2, \ldots, y^n, t$	$\{s, x^1\}$,
		$\{x^j, x_0^j, y_0^j, y^j\}, j \in [n]$,
		$\{y^j, x^{j+1}\}, j \in [n-1]$,
		$\{y^n, t\}$
L	$s, u^1, v^1, u^2, \ldots, v^m, t$	$\{s, u^1\}$,
		$\{u^i, \tilde{u}_L^i, \tilde{v}_L^i, v^i\}, i \in [m]$,
		$\{v^i, u^{i+1}\}, i \in [m-1]$,
		$\{v^m, t\}$
$\tilde{L}$	$s, u_L^1, v_L^1, u_R^1, v_R^1, u_L^2, \ldots, v_R^m, t$	$\{s, u_L^1\}$,
		$\{u_L^i, u_1^i, v_1^i, v_L^i\}, i \in [m]$,
		$\{v_L^i, u_R^i\}, i \in [m]$,
		$\{u_R^i, u_3^i, v_3^i, v_R^i\}, i \in [m]$,
		$\{v_R^i, u_L^{i+1}\}, i \in [m-1]$,
		$\{v_R^m, t\}$
X	$s, x^1, y^1, x^2, \ldots, y^n, t$	$\{s, x^1\}$,
		$\{x^j, x_1^j, y_1^j, y^j\}, j \in [n]$,
		$\{y^j, x^{j+1}\}, j \in [n-1]$,
		$\{y^n, t\}$
R	$s, u^1, v^1, u^2, \ldots, v^m, t$	$\{s, u^1\}$,
		$\{u^i, \tilde{u}_R^i, \tilde{v}_R^i, v^i\}, i \in [m]$,
		$\{v^i, u^{i+1}\}, i \in [m-1]$,
		$\{v^m, t\}$
$\tilde{R}$	$s, u_L^1, v_L^1, u_R^1, v_R^1, u_L^2, \ldots, v_R^m, t$	$\{s, u_L^1\}$,
		$\{u_L^i, u_2^i, v_2^i, v_L^i\}, i \in [m]$,
		$\{v_L^i, u_R^i\}, i \in [m]$,
		$\{u_R^i, u_4^i, v_4^i, v_R^i\}, i \in [m]$,
		$\{v_R^i, u_L^{i+1}\}, i \in [m-1]$,
		$\{v_R^m, t\}$
B	s, x^1, y^n, t	$\{s, x^1\}$, $\{x^j, y^j, u^i, v^i \mid j \in [n], i \in [m]\}$, $\{y^n, t\}$
$\tilde{B}$	$s, \tilde{u}_L^1, \tilde{v}_L^1, \tilde{u}_R^1, \tilde{v}_R^1, \tilde{u}_L^2, \ldots, \tilde{v}_R^m, t$	$\{s, \tilde{u}_L^1\}$,
		$\{\tilde{u}_L^i, u_L^i, v_L^i, \tilde{v}_L^i\}, i \in [m]$,
		$\{\tilde{v}_L^i, \tilde{u}_R^i\}, i \in [m]$,
		$\{\tilde{u}_R^i, u_R^i, v_R^i, \tilde{v}_R^i\}, i \in [m]$,
		$\{\tilde{v}_R^i, \tilde{u}_L^{i+1}\}, i \in [m-1]$,
		$\{\tilde{v}_R^m, t\}$
B_0	$s, x_0^1, y_0^1, x_0^2, \ldots, y_0^n, t$	$\{s, x_0^1\}$,
		$\{x_0^j, u_{\bar{\pi}(i,j)}^i, v_{\bar{\pi}(i,j)}^i, y_0^j \mid i \in \bar{P}_j\}, j \in [n]$,
		$\{y_0^n, t\}$
B_1	$s, x_1^1, y_1^1, x_1^2, \ldots, y_1^n, t$	$\{s, x_1^1\}$,
		$\{x_1^j, u_{\pi(i,j)}^i, v_{\pi(i,j)}^i, y_1^j \mid i \in P_j\}, j \in [n]$,
		$\{y_1^n, t\}$

(a) *if* $l_{i_1} = \bar{x}_j$, *then* $\mathcal{B}(x_0^j, B_0) < \mathcal{B}(u_L^i, \tilde{L})$, *and if* $l_{i_1} = x_j$, *then* $\mathcal{B}(x_1^j, B_1) < \mathcal{B}(u_L^i, \tilde{L})$,

(b) *if* $l_{i_2} = \bar{x}_j$, *then* $\mathcal{B}(x_0^j, B_0) < \mathcal{B}(u_L^i, \tilde{R})$, *and if* $l_{i_2} = x_j$, *then* $\mathcal{B}(x_1^j, B_1) < \mathcal{B}(u_L^i, \tilde{R})$,

(c) *if* $l_{i_3} = \bar{x}_j$, *then* $\mathcal{B}(x_0^j, B_0) < \mathcal{B}(u_R^i, \tilde{L})$, *and if* $l_{i_3} = x_j$, *then* $\mathcal{B}(x_1^j, B_1) < \mathcal{B}(u_R^i, \tilde{L})$,

(d) *if* $l_{i_4} = \bar{x}_j$, *then* $\mathcal{B}(x_0^j, B_0) < \mathcal{B}(u_R^i, \tilde{R})$, *and if* $l_{i_4} = x_j$, *then* $\mathcal{B}(x_1^j, B_1) < \mathcal{B}(u_R^i, \tilde{R})$.

6. *For every* $j \in [n]$,

(a) $\mathcal{B}(x^j, \bar{X}) < \mathcal{B}(x_0^j, B_0)$, *and*

(b) $\mathcal{B}(x^j, X) < \mathcal{B}(x_1^j, B_1)$.

Proof. We show every property by contradiction. More precisely, for every property, we assume it doesn't hold and then obtain an edge and a round such that the corresponding capacity constraint is violated, which contradicts the feasibility of block sequence $\mathcal{B}$.

Since every flow pair has demand 1, we may use 19 to argue about capacity constraints.

1, 3. We only show 1; the proofs for 3a and 3b are analogous. Suppose not. Then obtain $i \in [m]$ such that both $\mathcal{B}(u^i, L) \geq \mathcal{B}(x^1, B)$ and $\mathcal{B}(u^i, R) \geq \mathcal{B}(x^1, B)$. We show that the capacity constraint for edge (u^i, v^i) is violated for round $\mathcal{B}(x^1, B)$.

We have that

1. $\alpha_L((u^i, v^i), B_{\mathcal{B}(x^1,B)-1}) =$ active, since $b(u^i, L) \notin B_{\mathcal{B}(x^1,B)-1}$ and $(u^i, v^i) \in E(L^o)$,

2. $\alpha_R((u^i, v^i), B_{\mathcal{B}(x^1,B)-1}) =$ active, since $b(u^i, R) \notin B_{\mathcal{B}(x^1,B)-1}$ and $(u^i, v^i) \in E(R^o)$, and

3. $\alpha_B((u^i, v^i), B_{\mathcal{B}(x^1,B)}) =$ active, since $b(u^i, B) = b(x^1, B) \in B_{\mathcal{B}(x^1,B)}$ and $(u^i, v^i) \in E(B^u)$.

Hence

$$\begin{aligned}|\{P \in \mathcal{P}\ |&\alpha_P((u^i, v^i), B_{\mathcal{B}(x^1,B)-1}) = \text{active or} \\ &\alpha_P((u^i, v^i), B_{\mathcal{B}(x^1,B)}) = \text{active}\}| \geq |\{L, R, B\}| = 3 > 2 = c(u^i, v^i)\end{aligned}$$

2, 5, 6. We only show 2a; the proofs for 2b, 5a, 5b, 5c, 5d, 6a, and 6b are similar. Suppose not. Then obtain $i \in [m]$ such that $\mathcal{B}(\tilde{u}^i_L, \tilde{B}) \geq \mathcal{B}(u^i, L)$. We show that the capacity constraint for edge $(\tilde{u}^i_L, \tilde{v}^i_L)$ is violated for round $\mathcal{B}(u^i, L)$.

We have that

1. $\alpha_{\tilde{B}}((\tilde{u}^i_L, \tilde{v}^i_L), B_{\mathcal{B}(u^i,L)-1}) = \text{active}$, since $b(\tilde{u}^i_L, \tilde{B}) \notin B_{\mathcal{B}(u^i,L)-1}$ and $(\tilde{u}^i_L, \tilde{v}^i_L) \in E(\tilde{B}^o)$, and

2. $\alpha_L((\tilde{u}^i_L, \tilde{v}^i_L), B_{\mathcal{B}(u^i,L)}) = \text{active}$, since $b(\tilde{u}^i_L, L) = b(u^i, L) \in B_{\mathcal{B}(u^i,L)}$ and $(\tilde{u}^i_L, \tilde{v}^i_L) \in E(L^u)$.

Hence

$$|\{P \in \mathcal{P} \mid \alpha_P((\tilde{u}^i_L, \tilde{v}^i_L), B_{\mathcal{B}(u^i,L)-1}) = \text{active or} \\ \alpha_P((\tilde{u}^i_L, \tilde{v}^i_L), B_{\mathcal{B}(u^i,L)}) = \text{active}\}| \geq |\{\tilde{B}, L\}| = 2 > 1 = c(\tilde{u}^i_L, \tilde{v}^i_L)$$

4. Suppose not. Then obtain $j \in [n]$ such that both $\mathcal{B}(x^1, B) \geq \mathcal{B}(x^j, \bar{X})$ and $\mathcal{B}(x^1, B) \geq \mathcal{B}(x^j, X)$. We show that the capacity constraint for edge (x^j, y^j) is violated for round $\mathcal{B}(x^1, B)$.

We have that

1. $\alpha_B((x^j, y^j), B_{\mathcal{B}(x^1,B)-1}) = \text{active}$, since $b(x^j, B) = b(x^1, B) \notin B_{\mathcal{B}(x^1,B)-1}$ and $(x^j, y^j) \in E(B^o)$,

2. $\alpha_{\bar{X}}((x^j, y^j), B_{\mathcal{B}(x^1,B)}) = \text{active}$, since $b(x^j, \bar{X}) \notin B_{\mathcal{B}(x^1,B)}$ and $(x^j, y^j) \in E(\bar{X}^u)$, and

3. $\alpha_X((x^j, y^j), B_{\mathcal{B}(x^1,B)}) = \text{active}$, since $b(x^j, X) \notin B_{\mathcal{B}(x^1,B)}$ and $(x^j, y^j) \in E(X^u)$.

Hence

$$|\{P \in \mathcal{P} \mid \alpha_P((x^j, y^j), B_{\mathcal{B}(x^1,B)-1}) = \text{active or} \\ \alpha_P((x^j, y^j), B_{\mathcal{B}(x^1,B)}) = \text{active}\}| \geq |\{B, \bar{X}, X\}| = 3 > 2 = c(x^j, y^j)$$

□

We are now ready to prove Theorem 34.

Proof of Theorem [[thm:np-hardness-special-case.]] We show that there is a satisfying assignment σ for 4CNF formula C iff there is a feasible block sequence $\mathcal{B}$ for the corresponding update flow network G, which, by Corollary 21, is the case iff there is a feasible update sequence for G. We will choose σ, $\mathcal{B}$, respectively, such that σ assigns 1 to variable x_j iff $\mathcal{B}(x^j, \bar{X}) > \mathcal{B}(x^1, B)$.

Only-if part. Let $\mathcal{B}$ be a feasible block sequence for G. We define assignment σ as follows: For every variable x_j, we assign 1 to x_j iff $\mathcal{B}(x^j, \bar{X}) > \mathcal{B}(x^1, B)$. We now show that σ is a satisfying assignment for C.

Let $C_i = (l_{i_1} \vee l_{i_2} \vee l_{i_3} \vee l_{i_4})$ be a clause. We show that σ satisfies C_i by obtaining a literal that evaluates to 1.

Consider round $\mathcal{B}(x^1, B)$. By Lemma 37 1, $\mathcal{B}(x^1, B) > \mathcal{B}(u^i, L)$ or $\mathcal{B}(x^1, B) > \mathcal{B}(u^i, R)$. We only consider the former case $\mathcal{B}(x^1, B) > \mathcal{B}(u^i, L)$; the latter one is analogous.

By Lemma 37 2a, $\mathcal{B}(u^i, L) > \mathcal{B}(\tilde{u}^i_L, \tilde{B})$. By Lemma 37 3a, $\mathcal{B}(\tilde{u}^i_L, \tilde{B}) > \mathcal{B}(u^i_L, \tilde{L})$ or $\mathcal{B}(\tilde{u}^i_L, \tilde{B}) > \mathcal{B}(u^i_L, \tilde{R})$. We only consider the latter case $\mathcal{B}(\tilde{u}^i_L, \tilde{B}) > \mathcal{B}(u^i_L, \tilde{R})$; the former one is analogous.

Let x_j be the variable corresponding to literal l_{i_2}. We consider the cases $l_{i_2} = \bar{x}_j$ and $l_{i_2} = x_j$ separately.

Case $l_{i_2} = \bar{x}_j$. By Lemma 37 5b, $\mathcal{B}(u^i_L, \tilde{R}) > \mathcal{B}(x^j_0, B_0)$. By Lemma 37 6a, $\mathcal{B}(x^j_0, B_0) > \mathcal{B}(x^j, \bar{X})$. Putting everything together yields the following chain of inequalities:

$$\mathcal{B}(x^1, B) > \mathcal{B}(u^i, L) > \mathcal{B}(\tilde{u}^i_L, \tilde{B}) > \mathcal{B}(u^i_L, \tilde{R}) > \mathcal{B}(x^j_0, B_0) > \mathcal{B}(x^j, \bar{X})$$

Hence, by definition of our assignment, variable x_j is assigned 0. Hence literal $l_{i_2} = \bar{x}_j$ evaluates to 1.

Case $l_{i_2} = x_j$. By Lemma 37 5b, $\mathcal{B}(u^i_L, \tilde{R}) > \mathcal{B}(x^j_1, B_1)$. By Lemma 37 6b, $\mathcal{B}(x^j_1, B_1) > \mathcal{B}(x^j, X)$. Putting everything together yields the following chain of inequalities:

$$\mathcal{B}(x^1, B) > \mathcal{B}(u^i, L) > \mathcal{B}(\tilde{u}^i_L, \tilde{B}) > \mathcal{B}(u^i_L, \tilde{R}) > \mathcal{B}(x^j_1, B_1) > \mathcal{B}(x^j, X)$$

Hence, by Lemma 37 4, $\mathcal{B}(x^j, \bar{X}) > \mathcal{B}(x^1, B)$. Hence, by definition of our assignment, variable x_j is assigned 1. Hence literal $l_{i_2} = x_j$ evaluates to 1.

If part. Let σ be a satisfying assignment for C. We construct a feasible block sequence $\mathcal{B} = (\mathscr{B}_1, \ldots, \mathscr{B}_{11})$ for G as follows. The basic idea is to update blocks induced by

- variable vertices corresponding to variables that are assigned 1 and
- clause vertices corresponding to satisfied clauses

before we update block $b(x^1, B)$, and all other blocks afterwards. We now specify $\mathscr{B}_1, \ldots, \mathscr{B}_{11}$ in detail.

1. For every variable x_j, if x_j is assigned 1, we add block $b(x^j, X)$ to $\mathscr{B}_1$, otherwise we add $b(x^j, \bar{X})$. That is,

$$\mathscr{B}_1 = \{b(x^j, X) \mid \sigma(x_j) = 1\} \cup \{b(x^j, \bar{X} \mid \sigma(x_j) = 0\}.$$

2. For every variable x_j, if x_j is assigned 1, we add block $b(x_1^j, B_1)$ to $\mathscr{B}_2$, otherwise we add $b(x_0^j, B_0)$. That is,

$$\mathscr{B}_2 = \{b(x_1^j, B_1) \mid \sigma(x_j) = 1\} \cup \{b(x_0^j, B_0 \mid \sigma(x_j) = 0\}.$$

3. For every clause $C_i = (l_{i_1} \vee l_{i_2} \vee l_{i_3} \vee l_{i_4})$,

   (a) if l_{i_1} evaluates to 1, we add block $b(u_L^i, \tilde{L})$ to $\mathscr{B}_3$,

   (b) if l_{i_2} evaluates to 1, we add $b(u_L^i, \tilde{R})$,

   (c) if l_{i_3} evaluates to 1, we add $b(u_R^i, \tilde{L})$, and

   (d) if l_{i_4} evaluates to 1, we add $b(u_R^i, \tilde{R})$.

   That is,

$$\begin{aligned}\mathscr{B}_3 =& \{b(u_L^i, \tilde{L}) \mid \sigma(l_{i_1}) = 1\} \cup \{b(u_L^i, \tilde{R}) \mid \sigma(l_{i_2}) = 1\} \cup \\ & \{b(u_R^i, \tilde{L}) \mid \sigma(l_{i_3}) = 1\} \cup \{b(u_R^i, \tilde{R}) \mid \sigma(l_{i_4}) = 1\}.\end{aligned}$$

4. For every clause $C_i = (l_{i_1} \vee l_{i_2} \vee l_{i_3} \vee l_{i_4})$, if the left half $(l_{i_1} \vee l_{i_2})$ of C_i is satisfied, we add block $b(\tilde{u}_L^i, \tilde{B})$ to $\mathscr{B}_4$, and if the right half $(l_{i_3} \vee l_{i_4})$ is satisfied, we add $b(\tilde{u}_R^i, \tilde{B})$. That is,

$$\begin{aligned}\mathscr{B}_4 =& \{b(\tilde{u}_L^i, \tilde{B}) \mid \sigma(l_{i_1}) = 1 \text{ or } \sigma(l_{i_2}) = 1\} \cup \\ & \{b(\tilde{u}_R^i, \tilde{B}) \mid \sigma(l_{i_3}) = 1 \text{ or } \sigma(l_{i_4}) = 1\}.\end{aligned}$$

5. For every clause $C_i = (l_{i_1} \vee l_{i_2} \vee l_{i_3} \vee l_{i_4})$, if the left half $(l_{i_1} \vee l_{i_2})$ of C_i is satisfied, we add block $b(u^i, L)$ to $\mathscr{B}_5$, and if the right half $(l_{i_3} \vee l_{i_4})$ is satisfied, we add $b(u^i, R)$. That is,

$$\begin{aligned}\mathscr{B}_5 =& \{b(u^i, L) \mid \sigma(l_{i_1}) = 1 \text{ or } \sigma(l_{i_2}) = 1\} \cup \\ & \{b(u^i, R) \mid \sigma(l_{i_3}) = 1 \text{ or } \sigma(l_{i_4}) = 1\}.\end{aligned}$$

6. $\mathscr{B}_6 = \{b(x^1, B)\}$.

7. For every variable x_j, if x_j is assigned 0, we add block $b(x^j, X)$ to $\mathscr{B}_7$, otherwise we add $b(x^j, \bar{X})$. That is,

$$\mathscr{B}_7 = \{b(x^j, X) \mid \sigma(x_j) = 0\} \cup \{b(x^j, \bar{X} \mid \sigma(x_j) = 1\}.$$

8. For every variable x_j, if x_j is assigned 0, we add block $b(x_1^j, B_1)$ to $\mathscr{B}_8$, otherwise we add $b(x_0^j, B_0)$. That is,

$$\mathscr{B}_8 = \{b(x_1^j, B_1) \mid \sigma(x_j) = 0\} \cup \{b(x_0^j, B_0 \mid \sigma(x_j) = 1\}.$$

9. For every clause $C_i = (l_{i_1} \vee l_{i_2} \vee l_{i_3} \vee l_{i_4})$,

(a) if l_{i_1} evaluates to 0, we add block $b(u^i_L, \tilde{L})$ to $\mathscr{B}_9$,
(b) if l_{i_2} evaluates to 0, we add $b(u^i_L, \tilde{R})$,
(c) if l_{i_3} evaluates to 0, we add $b(u^i_R, \tilde{L})$, and
(d) if l_{i_4} evaluates to 0, we add $b(u^i_R, \tilde{R})$.

That is,

$$\begin{aligned}\mathscr{B}_9 =& \{b(u^i_L, \tilde{L}) \mid \sigma(l_{i_1}) = 0\} \cup \{b(u^i_L, \tilde{R}) \mid \sigma(l_{i_2}) = 0\} \cup \\ & \{b(u^i_R, \tilde{L}) \mid \sigma(l_{i_3}) = 0\} \cup \{b(u^i_R, \tilde{R}) \mid \sigma(l_{i_4}) = 0\}.\end{aligned}$$

10. For every clause $C_i = (l_{i_1} \vee l_{i_2} \vee l_{i_3} \vee l_{i_4})$, if the left half $(l_{i_1} \vee l_{i_2})$ of C_i is unsatisfied, we add block $b(\tilde{u}^i_L, \tilde{B})$ to $\mathscr{B}_{10}$, and if the right half $(l_{i_3} \vee l_{i_4})$ is unsatisfied, we add $b(\tilde{u}^i_R, \tilde{B})$. That is,

$$\begin{aligned}\mathscr{B}_{10} =& \{b(\tilde{u}^i_L, \tilde{B}) \mid \sigma(l_{i_1}) = 0 \text{ and } \sigma(l_{i_2}) = 0\} \cup \\ & \{b(\tilde{u}^i_R, \tilde{B}) \mid \sigma(l_{i_3}) = 0 \text{ and } \sigma(l_{i_4}) = 0\}.\end{aligned}$$

11. For every clause $C_i = (l_{i_1} \vee l_{i_2} \vee l_{i_3} \vee l_{i_4})$, if the left half $(l_{i_1} \vee l_{i_2})$ of C_i is unsatisfied, we add block $b(u^i, L)$ to $\mathscr{B}_{11}$, and if the right half $(l_{i_3} \vee l_{i_4})$ is unsatisfied, we add $b(u^i, R)$. That is,

$$\begin{aligned}\mathscr{B}_{11} =& \{b(u^i, L) \mid \sigma(l_{i_1}) = 0 \text{ and } \sigma(l_{i_2}) = 0\} \cup \\ & \{b(u^i, R) \mid \sigma(l_{i_3}) = 0 \text{ and } \sigma(l_{i_4}) = 0\}.\end{aligned}$$

By Remark 10, we may ignore all other blocks.

We now show that block sequence $\mathcal{B} = (\mathscr{B}_1, \ldots, \mathscr{B}_{11})$ is feasible by verifying that the capacity constraint is satisfied for every edge and every $\ell \in [11]$. Since every flow pair has demand 1, we

- may use remark 19 again to argue about capacity constraints, and
- only have to consider edges with capacity less than 10, that is, the number of flow pairs.

For every such edge e, we proceed as follows.

1. First, for every $\ell \in \{0, \ldots, 11\}$ and every flow pair P, we determine if e is on the transient (s,t)-path for P after updating all blocks in B_ℓ, that is, we determine if $\alpha_P(e, B_\ell) = \text{active}$.
2. Next, for every $\ell \in \{0, \ldots, 11\}$, we determine the set of flow pairs P such that $\alpha_P(e, B_\ell) = \text{active}$, that is, we determine the set $\mathcal{P}(e, \ell) := \{P \in \mathcal{P} \mid \alpha_P(e, B_\ell) = \text{active}\}$.
3. Then, for every $\ell \in [11]$, we determine the set $\mathcal{P}'(e, \ell) := \mathcal{P}(e, \ell - 1) \cup \mathcal{P}(e, \ell) = \{P \in \mathcal{P} \mid \alpha_P(e, B_{\ell-1}) = \text{active or } \alpha_P(e, B_\ell) = \text{active}\}$.
4. Finally, for every $\ell \in [11]$, we verify that the cardinality of the set $\mathcal{P}'(e, \ell)$ obtained in the previous step is at most $c(e)$.

(x^j, y^j) Let $j \in [n]$. Then edge (x^j, y^j) is used by flow pairs $\bar{X}, X, B$.
Since $(x^j, y^j) \in E(\bar{X}^u \setminus \bar{X}^o)$, by Lemma 15,

$$\alpha_{\bar{X}}((x^j, y^j), B_\ell) = \begin{cases} \text{active} & \text{if } \sigma(x_j) = 1 \text{ and } \ell \geq 7 \\ \text{active} & \text{if } \sigma(x_j) = 0 \text{ and } \ell \geq 1 \\ \text{inactive} & \text{otherwise.} \end{cases}$$

Since $(x^j, y^j) \in E(X^u \setminus X^o)$, by Lemma 15,

$$\alpha_X((x^j, y^j), B_\ell) = \begin{cases} \text{active} & \text{if } \sigma(x_j) = 1 \text{ and } \ell \geq 1 \\ \text{active} & \text{if } \sigma(x_j) = 0 \text{ and } \ell \geq 7 \\ \text{inactive} & \text{otherwise.} \end{cases}$$

Since $(x^j, y^j) \in E(B^o \setminus B^u)$ and $b(x^j, B) = b(x^1, B) \in \mathscr{B}_6$, by Lemma 15,

$$\alpha_B((x^j, y^j), B_\ell) = \begin{cases} \text{active} & \ell < 6 \\ \text{inactive} & \ell \geq 6. \end{cases}$$

Hence,

$$\mathcal{P}((x^j, y^j), \ell) = \begin{cases} \{B\} & \ell < 1 \\ \{X, B\} & \sigma(x_j) = 1 \text{ and } 1 \leq \ell < 6 \\ \{X\} & \sigma(x_j) = 1 \text{ and } \ell = 6 \\ \{\bar{X}, B\} & \sigma(x_j) = 0 \text{ and } 1 \leq \ell < 6 \\ \{\bar{X}\} & \sigma(x_j) = 0 \text{ and } \ell = 6 \\ \{\bar{X}, X\} & \ell \geq 7. \end{cases}$$

Hence,

$$\mathcal{P}'((x^j, y^j), \ell) = \begin{cases} \{X, B\} & \sigma(x_j) = 1 \text{ and } \ell < 7 \\ \{\bar{X}, B\} & \sigma(x_j) = 0 \text{ and } \ell < 7 \\ \{\bar{X}, X\} & \ell \geq 7. \end{cases}$$

Hence $|\mathcal{P}'((x^j, y^j), \ell)| = 2 = c(x^j, y^j)$ for every $\ell \in [11]$.

□ Repeat for other edges.

□

Chapter 5

Merging Flow Pairs

We now prove the Merging Lemma.

Let $G = (V, E, \mathcal{P}, s, t, c)$ be an update flow network with $|\mathcal{P}| \geq 2$, and let $F, F' \in \mathcal{P}$ and $v_F, v_{F'} \in V$ such that they satisfy properties 1, 2, and 3 (see Figure 5.1). We construct an update flow network $\tilde{G} = (\tilde{V}, \tilde{E}, \tilde{\mathcal{P}}, s, t, \tilde{c})$ with $|\tilde{\mathcal{P}}| = |\mathcal{P}| - 1$ such that there is a feasible block sequence $\mathcal{B} = (\mathscr{B}_1, \ldots, \mathscr{B}_\ell)$ for G iff there is a feasible block sequence $\tilde{\mathcal{B}} = (\tilde{\mathscr{B}}_1, \ldots, \tilde{\mathscr{B}}_\ell)$ for $\tilde{G}$ as follows.

5.1 The Construction

Intuitively, we merge flow pairs F and F' into a single flow pair $\tilde{F}$ by concatenating F and F'. More precisely, $\tilde{F}$ will be the union of F and F' except that we replace edges (v_F, t) and $(s, v_{F'})$ by edge $(v_F, v_{F'})$ (see Figure 5.2 for an illustration). More formally, we define flow pair $\tilde{F}$ as follows:

$$\begin{aligned}
\tilde{E}(\tilde{F}^o) &= (E(F^o) \setminus \{(v_F, t)\}) \cup (E(F'^o) \setminus \{(s, v_{F'})\}) \cup \{(v_F, v_{F'})\} \\
\tilde{E}(\tilde{F}^u) &= (E(F^u) \setminus \{(v_F, t)\}) \cup (E(F'^u) \setminus \{(s, v_{F'})\}) \cup \{(v_F, v_{F'})\} \\
\tilde{V}(\tilde{F}^o) &= \tilde{V}(\tilde{E}(\tilde{F}^o)) \\
\tilde{V}(\tilde{F}^u) &= \tilde{V}(\tilde{E}(\tilde{F}^u)) \\
\tilde{d}_{\tilde{F}} &= d_F
\end{aligned}$$

Update flow network $\tilde{G} = (\tilde{V}, \tilde{E}, \tilde{\mathcal{P}}, s, t, \tilde{c})$ is defined as follows:

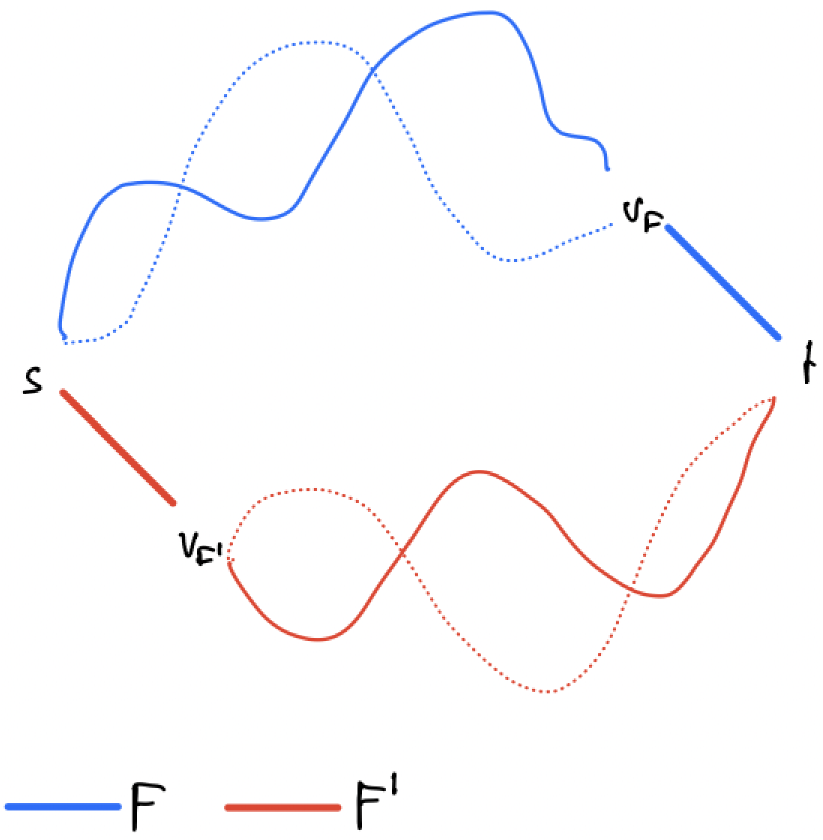

Figure 5.1: Flow pairs F and F' in update flow network G

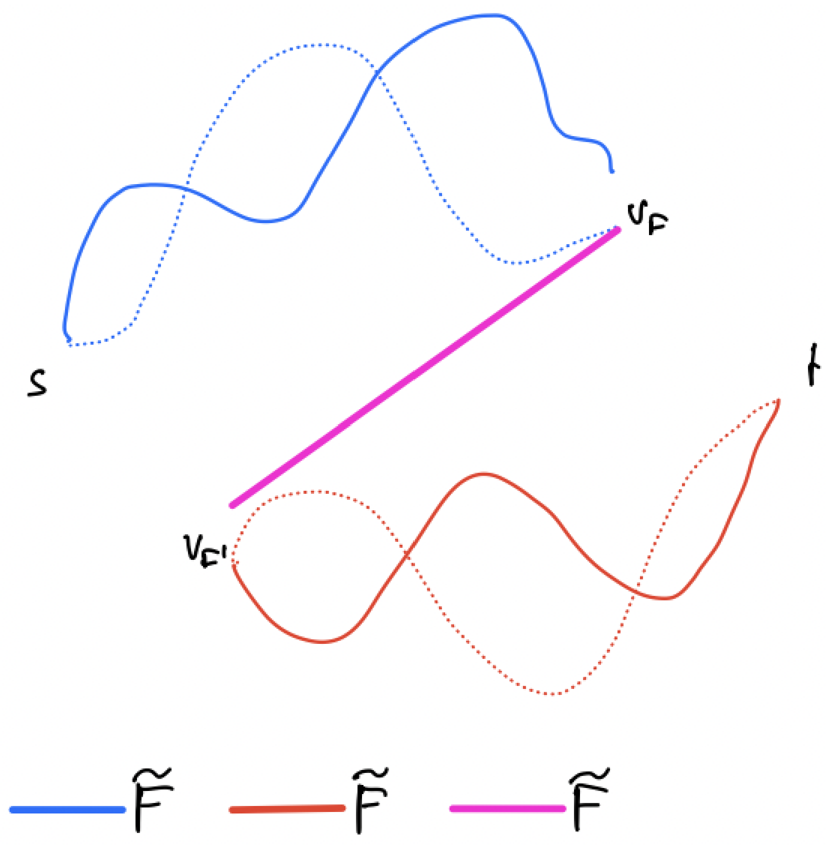

Figure 5.2: Flow pair $\tilde{F}$ in update flow network $\tilde{G}$

$$
\begin{aligned}
\tilde{\mathcal{P}} &= \mathcal{P} \setminus \{F, F'\} \cup \{\tilde{F}\} \\
\tilde{V} &= \bigcup_{\tilde{P} \in \tilde{\mathcal{P}}} \tilde{V}(\tilde{P}^o \cup \tilde{P}^u) \\
\tilde{E} &= \bigcup_{\tilde{P} \in \tilde{\mathcal{P}}} \tilde{E}(\tilde{P}^o \cup \tilde{P}^u) \\
\tilde{c}(\tilde{e}) &= \begin{cases} \sum_{\tilde{P} \in \tilde{\mathcal{P}} : \tilde{e} \in \tilde{E}(\tilde{P}^o \cup \tilde{P}^u)} \tilde{d}_{\tilde{P}} & \text{if } \tilde{e} = (v_F, v_{F'}) \\ c(\tilde{e}) & \text{otherwise} \end{cases}
\end{aligned}
$$

Let us quickly verify that $\tilde{G}$ is a feasible update flow network.

Let $\tilde{P} \in \tilde{\mathcal{P}}$. If $\tilde{P} \neq \tilde{F}$, then $\tilde{P} \in \mathcal{P}$ and hence, by feasibility of update flow network G, both $\tilde{P}^o$ and $\tilde{P}^u$ are (s,t)-paths in $\tilde{G}$ and $\tilde{P}$ forms a DAG. Now suppose $\tilde{P} = \tilde{F}$. By feasibility of G and construction of $\tilde{F}$, $\tilde{F}^o$ ($\tilde{F}^u$) comprises the (s, v_F)-path in F^o (F^u), edge $(v_F, v_{F'})$, and the $(v_{F'}, t)$-path in F'^o (F'^u), and hence forms an (s,t)-path. Moreover, since, again by feasibility of G, both F and F' form DAGs, and edge $(v_F, v_{F'})$ does not introduce a cycle, as F and F' have no common vertices other than s, t by assumption, $\tilde{F}$ forms a DAG.

Using Lemma 20, we will show that all capacity constraits are satisfied for both the old flow network and the updated flow network in the if part of the proof of the Merging Lemma.

We denote notations such as $b(v, P)$, B_i, and $\alpha_P(e, B)$ referring to update flow network $\tilde{G}$ by $\tilde{b}(v, P)$, $\tilde{B}_i$, and $\tilde{\alpha}_P(e, B)$.

5.2 The Proof

Our goal is to show that there is a feasible block sequence $\mathcal{B} = (\mathscr{B}_1, \ldots, \mathscr{B}_\ell)$ for G iff there is a feasible block sequence $\tilde{\mathcal{B}} = (\tilde{\mathscr{B}}_1, \ldots, \tilde{\mathscr{B}}_\ell)$ for $\tilde{G}$. We will choose $\mathcal{B}, \tilde{\mathcal{B}}$, respectively, such that, for every block b contained in both G and $\tilde{G}$, b is updated in round $i \in [\ell]$ in $\mathcal{B}$ iff it is updated in round i in $\tilde{\mathcal{B}}$, that is, $\mathcal{B}(b) = \tilde{\mathcal{B}}(b)$. The key insight is that it is indeed sufficient to consider such blocks.

Lemma 38.

1. *Let $\tilde{u} \in \tilde{V}(\tilde{F}^o \cup \tilde{F}^u) \setminus \{v_F\}$. Then:*
   - *(a) If $\tilde{u} \in V(F^o \cup F^u)$, then $\tilde{b}(\tilde{u}, \tilde{F}) = b(\tilde{u}, F)$.*
   - *(b) If $\tilde{u} \in V(F'^o \cup F'^u) \setminus \{s\}$, then $\tilde{b}(\tilde{u}, \tilde{F}) = b(\tilde{u}, F')$.*
2. *For every $\tilde{P} \in \tilde{\mathcal{P}} \setminus \{\tilde{F}\}$ and every $\tilde{u} \in \tilde{V}(\tilde{P}^o \cup \tilde{P}^u)$, $\tilde{b}(\tilde{u}, \tilde{P}) = b(\tilde{u}, \tilde{P})$.*

☐ I'm pretty sure we have to exclude $\tilde{u} = s$ in 1a.

Remark 39. The proof is very technical and tedious–and hence omitted for now–and I hope we can come up with a better characterization of blocks (see `../README.org`) which significantly simplifies the proof.

Corollary 40.

1. *For every block $\tilde{b} \in \tilde{B}(\tilde{G}) \setminus \{\{v_F, v_{F'}\}\}$, $\tilde{b} \in B(G)$.*

2. *For every block $b \in B(G) \setminus \{\{v_F, t\}, \{s, v_{F'}\}\}$, $b \in \tilde{B}(\tilde{G})$.*

Proof.

1. Let $\tilde{b} \in \tilde{B}(\tilde{G}) \setminus \{\{v_F, v_{F'}\}\}$, $\tilde{P} = \tilde{P}(\tilde{b})$, and $\tilde{u} = \tilde{\mathcal{S}}(\tilde{b})$. If $\tilde{P} = \tilde{F}$, then, by assumption, $\tilde{u} \neq v_F$ and hence, by construction of $\tilde{F}$ and Lemma 38 1, $\tilde{b} = b(\tilde{u}, F) \in B(G)$ or $\tilde{b} = b(\tilde{u}, F') \in B(G)$. If $\tilde{P} \neq \tilde{F}$, then, by Lemma 38 2, $\tilde{b} = b(\tilde{u}, \tilde{P}) \in B(G)$.

2. Let $b \in B(G) \setminus \{\{v_F, t\}, \{s, v_{F'}\}\}$, $P = P(b)$, and $u = \mathcal{S}(b)$. If $P = F$, then, by assumption, $u \neq v_F$ and hence, by construction of $\tilde{F}$ and Lemma 38 1a, $b = \tilde{b}(u, \tilde{F}) \in \tilde{B}(\tilde{G})$. If $P = F'$, then, by assumption, $u \notin \{v_F, s\}$ and hence, by construction of $\tilde{F}$ and Lemma 38 1b, $b = \tilde{b}(u, \tilde{F}) \in \tilde{B}(\tilde{G})$. If $P \in \mathcal{P} \setminus \{F, F'\}$, then $P \in \tilde{\mathcal{P}} \setminus \{\tilde{F}\}$ and hence, by Lemma 38 2, $b = \tilde{b}(u, P) \in \tilde{B}(\tilde{G})$. □

To show that block sequences $\mathcal{B}, \tilde{\mathcal{B}}$ as chosen above are feasible, we will verify that capacity constraint 2.1 is satisfied for every edge $e \in E$, $\tilde{e} \in \tilde{E}$, respectively, and every $i \in [\ell]$. We now show that for every edge $\tilde{e}$ other than $(v_F, t), (s, v_{F'}), (v_F, v_{F'})$ and every $i \in [\ell]$, $\tilde{e}$ is on some transient (s,t)-path in $\tilde{G}$ after updating all blocks in $\tilde{B}_i$ iff it is on some transient (s,t)-path in G after updating all blocks in B_i.

Lemma 41. *Let $\mathcal{B} = (\mathscr{B}_1, \dots, \mathscr{B}_\ell)$ be a block sequence for G and $\tilde{\mathcal{B}} = (\tilde{\mathscr{B}}_1, \dots, \tilde{\mathscr{B}}_\ell)$ be a block sequence for $\tilde{G}$ such that for every block b contained in both G and $\tilde{G}$, $\mathcal{B}(b) = \tilde{\mathcal{B}}(b)$. Moreover, let $(\tilde{u}, \tilde{v}) \in \tilde{E} \setminus \{(v_F, t), (s, v_{F'}), (v_F, v_{F'})\}$ and $i \in [\ell]$. Finally, let $\tilde{P} \in \tilde{\mathcal{P}}$ such that $(\tilde{u}, \tilde{v}) \in \tilde{E}(\tilde{P}^o \cup \tilde{P}^u)$. Then:*

1. *If $\tilde{P} = \tilde{F}$, then $\tilde{\alpha}_{\tilde{P}}((\tilde{u}, \tilde{v}), \tilde{B}_i) = \text{active}$ iff either $\alpha_F((\tilde{u}, \tilde{v}), B_i) = \text{active}$ or $\alpha_{F'}((\tilde{u}, \tilde{v}), B_i) = \text{active}$.*

2. *If $\tilde{P} \neq \tilde{F}$, then $\tilde{\alpha}_{\tilde{P}}((\tilde{u}, \tilde{v}), \tilde{B}_i) = \alpha_{\tilde{P}}((\tilde{u}, \tilde{v}), B_i)$.*

Proof. Let $\mathcal{B} = (\mathscr{B}_1, \dots, \mathscr{B}_\ell)$ be a block sequence for G and $\tilde{\mathcal{B}} = (\tilde{\mathscr{B}}_1, \dots, \tilde{\mathscr{B}}_\ell)$ be a block sequence for $\tilde{G}$ such that for every block b satisfying both $b \in B(G)$ and $b \in \tilde{B}(\tilde{G})$, $\mathcal{B}(b) = \tilde{\mathcal{B}}(b)$. Let $(\tilde{u}, \tilde{v}) \in \tilde{E} \setminus \{(v_F, t), (s, v_{F'}), (v_F, v_{F'})\}$ and $i \in [\ell]$. Let $\tilde{P} \in \tilde{\mathcal{P}}$ such that $(\tilde{u}, \tilde{v}) \in \tilde{E}(\tilde{P}^o \cup \tilde{P}^u)$.

1. Suppose $\tilde{P} = \tilde{F}$. By definition of $\tilde{F}$ and since $(\tilde{u}, \tilde{v}) \in \tilde{E} \backslash \{(v_F, t), (s, v_{F'}), (v_F, v_{F'})\}$, $(\tilde{u}, \tilde{v}) \in \tilde{E}(\tilde{F}^o)$ iff $(\tilde{u}, \tilde{v}) \in E(F^o)$ or $(\tilde{u}, \tilde{v}) \in E(F'^o)$. Similarly, $(\tilde{u}, \tilde{v}) \in \tilde{E}(\tilde{F}^u)$ iff $(\tilde{u}, \tilde{v}) \in E(F^u)$ or $(\tilde{u}, \tilde{v}) \in E(F'^u)$. We show $\tilde{\alpha}_{\tilde{F}}((\tilde{u}, \tilde{v}), \tilde{B}_i) =$ active iff $\alpha_F((\tilde{u}, \tilde{v}), B_i) =$ active or $\alpha_{F'}((\tilde{u}, \tilde{v}), B_i) =$ active. Notice that this implies 1, since, by assumption, F, F' are edge-disjoint: Otherwise, either

1. F and F' have a common vertex other than s, t, or

2. $F^o \cup F^u$ and $F'^o \cup F'^u$ both consist of the single edge (s, t), in which case $v_F = s$ and $v_{F'} = t$, which contradicts that $(v_F, v_{F'}) \notin E$.

We first show the if part. Suppose $\tilde{\alpha}_{\tilde{F}}((\tilde{u}, \tilde{v}), \tilde{B}_i) =$ active. By assumption, $(\tilde{u}, \tilde{v}) \in \tilde{E}(\tilde{F}^o)$ or $(\tilde{u}, \tilde{v}) \in \tilde{E}(\tilde{F}^u)$. Hence $(\tilde{u}, \tilde{v}) \in E(F^o)$ or $(\tilde{u}, \tilde{v}) \in E(F^u)$ or $(\tilde{u}, \tilde{v}) \in E(F'^o)$ or $(\tilde{u}, \tilde{v}) \in E(F'^u)$. We only consider case $(\tilde{u}, \tilde{v}) \in E(F^o)$; case $(\tilde{u}, \tilde{v}) \in E(F^u)$ is similar, and cases $(\tilde{u}, \tilde{v}) \in E(F'^o)$, $(\tilde{u}, \tilde{v}) \in E(F'^u)$ are analogous to cases $(\tilde{u}, \tilde{v}) \in E(F^o)$, $(\tilde{u}, \tilde{v}) \in E(F^u)$, respectively.

Suppose $(\tilde{u}, \tilde{v}) \in E(F^o)$. Hence $(\tilde{u}, \tilde{v}) \in \tilde{E}(\tilde{F}^o)$. Hence $\tilde{b}(\tilde{u}, \tilde{F}) \in \tilde{B}_i$. Moreover, by Lemma 38 1a, $\tilde{b}(\tilde{u}, \tilde{F}) = b(\tilde{u}, F)$. Hence, by assumption, $b(\tilde{u}, F) \in B_i$. Thus $\alpha_F((\tilde{u}, \tilde{v}), B_i) =$ active.

We now show the only-if part. Suppose $\alpha_F((\tilde{u}, \tilde{v}), B_i) =$ active or $\alpha_{F'}((\tilde{u}, \tilde{v}), B_i) =$ active. We only consider the former case; the latter one is analogous.

Suppose $\alpha_F((\tilde{u}, \tilde{v}), B_i) =$ active. Hence $(\tilde{u}, \tilde{v}) \in E(F^o)$ or $(\tilde{u}, \tilde{v}) \in E(F^u)$. We again only consider the former case; the latter one is similar.

Suppose $(\tilde{u}, \tilde{v}) \in E(F^o)$. Hence $b(\tilde{u}, F) \in B_i$. Moreover, by Lemma 38 1a, $b(\tilde{u}, F) = \tilde{b}(\tilde{u}, \tilde{F})$. Hence, by assumption, $\tilde{b}(\tilde{u}, \tilde{F}) \in \tilde{B}_i$. Moreover, $(\tilde{u}, \tilde{v}) \in \tilde{E}(\tilde{F}^o)$. Thus $\tilde{\alpha}_{\tilde{F}}((\tilde{u}, \tilde{v}), \tilde{B}_i) =$ active.

2. Suppose $\tilde{P} \neq \tilde{F}$. By definition of $\tilde{G}$, $\tilde{P} \in \mathcal{P} \setminus \{F, F'\}$ and hence both $(\tilde{u}, \tilde{v}) \in \tilde{E}(\tilde{P}^o)$ iff $(\tilde{u}, \tilde{v}) \in E(\tilde{P}^o)$ and $(\tilde{u}, \tilde{v}) \in \tilde{E}(\tilde{P}^u)$ iff $(\tilde{u}, \tilde{v}) \in E(\tilde{P}^u)$. Hence $\tilde{b}(\tilde{u}, \tilde{P}) \in \tilde{B}(\tilde{G})$ and $\tilde{b}(\tilde{u}, \tilde{P}) \in B(G)$. Hence, by assumption, $\tilde{b}(\tilde{u}, \tilde{P}) \in \tilde{B}_i$ iff $\tilde{b}(\tilde{u}, \tilde{P}) \in B_i$, and, by Lemma 38 2, $\tilde{b}(\tilde{u}, \tilde{P}) = b(\tilde{u}, \tilde{P})$. Hence $\tilde{b}(\tilde{u}, \tilde{P}) \in \tilde{B}_i$ iff $b(\tilde{u}, \tilde{P}) \in B_i$. The claim now follows by definitions of $\tilde{\alpha}_{\tilde{P}}, \alpha_{\tilde{P}}$. □

Lemma 42. *Let $\mathcal{B} = (\mathscr{B}_1, \ldots, \mathscr{B}_\ell)$ be a block sequence for G and $\tilde{\mathcal{B}} = (\tilde{\mathscr{B}}_1, \ldots, \tilde{\mathscr{B}}_\ell)$ be a block sequence for $\tilde{G}$ such that for every block b contained in both G and $\tilde{G}$, $\mathcal{B}(b) = \tilde{\mathcal{B}}(b)$. Moreover, let $(\tilde{u}, \tilde{v}) \in \tilde{E} \setminus \{(v_F, t), (s, v_{F'}), (v_F, v_{F'})\}$ and $i \in [\ell]$. Then*

$$\sum_{\tilde{P} \in \tilde{\mathcal{P}} : \tilde{\alpha}_{\tilde{P}}((\tilde{u}, \tilde{v}), \tilde{B}_{i-1}) = \text{active } \textit{or } \tilde{\alpha}_{\tilde{P}}((\tilde{u}, \tilde{v}), \tilde{B}_i) = \text{active}} \tilde{d}_{\tilde{P}} =$$
$$\sum_{\tilde{P} \in \mathcal{P} : \alpha_{\tilde{P}}((\tilde{u}, \tilde{v}), B_{i-1}) = \text{active } \textit{or } \alpha_{\tilde{P}}((\tilde{u}, \tilde{v}), B_i) = \text{active}} d_{\tilde{P}}.$$

Proof. Let $\mathcal{B} = (\mathscr{B}_1, \ldots, \mathscr{B}_\ell)$ be a block sequence for G and $\tilde{\mathcal{B}} = (\tilde{\mathscr{B}}_1, \ldots, \tilde{\mathscr{B}}_\ell)$ be a block sequence for $\tilde{G}$ such that for every block b satisfying both $b \in B(G)$ and $b \in \tilde{B}(\tilde{G})$, $\mathcal{B}(b) = \tilde{\mathcal{B}}(b)$. Let $(\tilde{u}, \tilde{v}) \in \tilde{E} \setminus \{(v_F, t), (s, v_{F'}), (v_F, v_{F'})\}$ and $i \in [\ell]$. By definition of $\tilde{G}$, $\tilde{\mathcal{P}} = \mathcal{P} \setminus \{F, F'\} \cup \{\tilde{F}\}$, $\tilde{d}_{\tilde{F}} = d_F$, and $\tilde{d}_{\tilde{P}} = d_{\tilde{P}}$ for every $\tilde{P} \in \tilde{\mathcal{P}} \setminus \{\tilde{F}\}$. Moreover, by assumption, $d_F = d_{F'}$. Hence, by Lemma 41, we have

$$\begin{aligned}
&\sum_{\tilde{P}\in\tilde{\mathcal{P}}:\tilde{\alpha}_{\tilde{P}}((\tilde{u},\tilde{v}),\tilde{B}_{i-1})=\text{active or }\tilde{\alpha}_{\tilde{P}}((\tilde{u},\tilde{v}),\tilde{B}_i)=\text{active}} \tilde{d}_{\tilde{P}} = \\
&\sum_{\tilde{P}\in\tilde{\mathcal{P}}\setminus\{\tilde{F}\}:\tilde{\alpha}_{\tilde{P}}((\tilde{u},\tilde{v}),\tilde{B}_{i-1})=\text{active or }\tilde{\alpha}_{\tilde{P}}((\tilde{u},\tilde{v}),\tilde{B}_i)=\text{active}} \tilde{d}_{\tilde{P}} \\
+&\sum_{\tilde{P}\in\{\tilde{F}\}:\tilde{\alpha}_{\tilde{P}}((\tilde{u},\tilde{v}),\tilde{B}_{i-1})=\text{active or }\tilde{\alpha}_{\tilde{P}}((\tilde{u},\tilde{v}),\tilde{B}_i)=\text{active}} \tilde{d}_{\tilde{P}} = \\
&\sum_{\tilde{P}\in\mathcal{P}\setminus\{F,F'\}:\alpha_{\tilde{P}}((\tilde{u},\tilde{v}),B_{i-1})=\text{active or }\alpha_{\tilde{P}}((\tilde{u},\tilde{v}),B_i)=\text{active}} d_{\tilde{P}} \\
+&\sum_{\tilde{P}\in\{F,F'\}:\alpha_{\tilde{P}}((\tilde{u},\tilde{v}),B_{i-1})=\text{active or }\alpha_{\tilde{P}}((\tilde{u},\tilde{v}),B_i)=\text{active}} d_{\tilde{P}} = \\
&\sum_{\tilde{P}\in\mathcal{P}:\alpha_{\tilde{P}}((\tilde{u},\tilde{v}),B_{i-1})=\text{active or }\alpha_{\tilde{P}}((\tilde{u},\tilde{v}),B_i)=\text{active}} d_{\tilde{P}}.
\end{aligned}$$

□

We are now ready to prove the Merging Lemma.

Proof of Lemma [[lem:merging-flow-pairs.]] We show that there is a feasible block sequence $\mathcal{B} = (\mathscr{B}_1, \ldots, \mathscr{B}_\ell)$ for G iff there is a feasible block sequence $\tilde{\mathcal{B}} = (\tilde{\mathscr{B}}_1, \ldots, \tilde{\mathscr{B}}_\ell)$ for $\tilde{G}$.

If part. Let $\mathcal{B} = (\mathscr{B}_1, \ldots, \mathscr{B}_\ell)$ be a feasible block sequence for G. We define block sequence $\tilde{\mathcal{B}} = (\tilde{\mathscr{B}}_1, \ldots, \tilde{\mathscr{B}}_\ell)$ for $\tilde{G}$ as follows. By Remark 10, we may ignore block $\{v_F, v_{F'}\}$. For every other block $\tilde{b} \in \tilde{B}(\tilde{G}) \setminus \{\{v_F, v_{F'}\}\}$, we define $\tilde{\mathcal{B}}(\tilde{b}) = \mathcal{B}(\tilde{b})$. Notice that if $\tilde{b} \in \tilde{B}(\tilde{G}) \setminus \{\{v_F, v_{F'}\}\}$, then, by Lemma 40 1, $\tilde{b} \in B(G)$ and hence $\mathcal{B}(\tilde{b})$ is defined.

We now show that $\tilde{\mathcal{B}}$ is feasible. Let $(\tilde{u}, \tilde{v}) \in \tilde{E}$ and $i \in [\ell]$. We show that the capacity constraint for $(\tilde{u}, \tilde{v})$ and i is satisfied.

If $(\tilde{u}, \tilde{v}) = (v_F, v_{F'})$, then, by definition of $\tilde{G}$,

$$\tilde{c}(\tilde{u}, \tilde{v}) \geq \sum_{\tilde{P}\in\tilde{\mathcal{P}}:(\tilde{u},\tilde{v})\in E(\tilde{P}^o\cup\tilde{P}^u)} d_{\tilde{P}}$$

and hence the capacity constraint is satisfied.

Now suppose $(\tilde{u},\tilde{v}) \neq (v_F, v_{F'})$. Hence, by definition of $\tilde{G}$, $(\tilde{u},\tilde{v}) \in E$ and $\tilde{c}(\tilde{u},\tilde{v}) = c(\tilde{u},\tilde{v})$. If $(\tilde{u},\tilde{v}) \in \{(v_F,t),(s,v_{F'})\}$, then by assumption 3c, the capacity constraint is satisfied.

Now suppose $(\tilde{u},\tilde{v}) \notin \{(v_F,t),(s,v_{F'}),(v_F,v_{F'})\}$. Hence, by Lemma 42 and since block sequence $\mathcal{B}$ is feasible, we have

$$\sum_{\tilde{P}\in\tilde{\mathcal{P}}:\tilde{\alpha}_{\tilde{P}}((\tilde{u},\tilde{v}),\tilde{B}_{i-1})=\text{active or }\tilde{\alpha}_{\tilde{P}}((\tilde{u},\tilde{v}),\tilde{B}_i)=\text{active}} \tilde{d}_{\tilde{P}} =$$
$$\sum_{\tilde{P}\in\mathcal{P}:\alpha_{\tilde{P}}((\tilde{u},\tilde{v}),B_{i-1})=\text{active or }\alpha_{\tilde{P}}((\tilde{u},\tilde{v}),B_i)=\text{active}} d_{\tilde{P}} \leq$$
$$c(\tilde{u},\tilde{v}) = \tilde{c}(\tilde{u},\tilde{v}).$$

Only-if part. Let $\tilde{\mathcal{B}} = (\tilde{\mathscr{B}}_1,\ldots,\tilde{\mathscr{B}}_\ell)$ be a feasible block sequence for $\tilde{G}$. We define block sequence $\mathcal{B} = (\mathscr{B}_1,\ldots,\mathscr{B}_\ell)$ for G as follows. By Remark 10, we may ignore blocks $\{v_F,t\},\{s,v_{F'}\}$. For every other block $b \in B(G) \setminus \{\{v_F,t\},\{s,v_{F'}\}\}$, we define $\mathcal{B}(b) = \tilde{\mathcal{B}}(b)$. Notice that if $b \in B(G)\setminus\{\{v_F,t\},\{s,v_{F'}\}\}$, then, by Lemma 40 2, $b \in \tilde{B}(\tilde{G})$ and hence $\tilde{\mathcal{B}}(b)$ is defined.

We now show that $\mathcal{B}$ is feasible. Let $(u,v) \in E$ and $i \in [\ell]$. We show that the capacity constraint for (u,v) and i is satisfied.

If $(u,v) \in \{(v_F,t),(s,v_{F'})\}$, then, by assumption 3c, the capacity constraint is satisfied.

Now suppose $(u,v) \notin \{(v_F,t),(s,v_{F'})\}$. Hence, by assumption 3a, $(u,v) \notin \{(v_F,t),(s,v_{F'}),(v_F,v_{F'})\}$. Hence, by definition of $\tilde{G}$, $(u,v) \in \tilde{E}$ and $c(u,v) = \tilde{c}(u,v)$. Hence, by Lemma 42 and since block sequence $\tilde{\mathcal{B}}$ is feasible, we have

$$\sum_{P\in\mathcal{P}:\alpha_P((u,v),B_{i-1})=\text{active or }\alpha_P((u,v),B_i)=\text{active}} d_P =$$
$$\sum_{P\in\tilde{\mathcal{P}}:\tilde{\alpha}_P((u,v),\tilde{B}_{i-1})=\text{active or }\tilde{\alpha}_P((u,v),\tilde{B}_i)=\text{active}} \tilde{d}_P \leq$$
$$\tilde{c}(u,v) = c(u,v).$$

□

Remark 43. Let G, F, F' and $\tilde{G}, \tilde{F}$ be as specified in the proof of the Merging Lemma. Then notice that:

1. For every flow pair $P \in \mathcal{P} \setminus \{F, F'\}$, if properties 1, 2, and 3 are satisfied for both F, P and F', P in G, then they are also satisfied for $\tilde{F}, P$ in $\tilde{G}$.

2. For every two flow pairs $P, P' \in \mathcal{P} \setminus \{F, F'\}$, if properties 1, 2, and 3 are satisfied for P, P' in G, then they are also satisfied for P, P' in $\tilde{G}$.

Chapter 6

NP-Hardness for the General Case

We are finally ready to prove Theorem 33.

Proof of Theorem [[thm:np-hardness-k-eq-3.]] Let C be a 4CNF formula and G be the corresponding update flow network as specified in the proof of Theorem 34. Then G comprises 10 flow pairs $\bar{X}, L, \tilde{L}, X, R, \tilde{R}, B, \tilde{B}, B_0, B_1$, each with demand 1.

By definition, flow pairs $\bar{X}, L, \tilde{L}$ pairwise satisfy assumptions 1, 2, and 3 of the Merging Lemma (confirm Table 4.1). Hence, by Remark 43 1, we may apply the Merging Lemma twice to obtain from G an update flow network G_8 with 8 flow pairs.

Similarly, flow pairs $X, R, \tilde{R}$ pairwise satisfy assumptions 1, 2, and 3 of the Merging Lemma in G. By Remark 43 2, they also satisfy these assumptions in G_8. Hence, again by Remark 43 1, we may apply the Merging Lemma twice to obtain from G_8 an update flow network G_6 with 6 flow pairs.

Finally, as flow pairs $B, \tilde{B}, B_0, B_1$ again pairwise satisfy all three assumptions, we apply the Merging Lemma three times to obtain from G_6 an update flow network G_3 with 3 flow pairs.

Putting everything together, we have that there is a satisfying assignment for 4CNF formula C iff there is a feasible block sequence for update flow network G iff there is a feasible block sequence for G_8 iff there is a feasible block sequence for G_6 iff there is a feasible block sequence for G_3, which, by Corollary 21, is the case iff there is a feasible update sequence for G_3. □

Part III

Bounded Treewidth

The goal of this section is to prove the following theorem.

Theorem 44. *There is an algorithm that, given an update flow network G with k flow pairs and a nice tree decomposition of G with $O(n)$ nodes and width $\ell-1$, either determines that there is no feasible block sequence for G or finds such a sequence in time $O((k\ell)!\,k\ell n^2)$.*

☐ Define tree decomposition of an update flow network.

Before we will prove how to find a feasible block sequence in section _, we will show how to decide if there is such a sequence in section _.

Chapter 7

Decision Problem

Theorem 45. *There is an algorithm that, given an update flow network G with k flow pairs and a nice tree decomposition of G with $O(n)$ nodes and width $\ell - 1$, decides whether there is a feasible block sequence for G in time $O((k\ell)!\, k\ell n^2)$.*

Let G be an update flow network with k flow pairs and $(T, \{X_x \mid x \in V(T)\})$ be a nice tree decomposition of G with $O(n)$ nodes and width $\ell - 1$. For a node $x \in V(T)$, we denote by V_x the union of all bags X_y with $y = x$ or y is a descendant of x in T, and by $G_x = (V_x, E(V_x))$ the subgraph of G induced by V_x.

☐ Remove G_x?

The dynamic programming algorithm proceeds as follows. For every node $x \in V(T)$, the algorithm computes a table D_x which contains an entry $D_x[\pi]$ for every X_x-permutation π. Every entry $D_x[\pi]$ is a single bit indicating whether X_x-permutation π can be extended to a V_x-feasible permutation. This is done in bottom-up order, that is, the table of node x is computed after the tables of all descendants of x have been computed. In particular, to compute the table of node x, the algorithm uses the tables of the children of x. After all tables have been computed, the algorithm can decide if there is a feasible block sequence for update flow network G as follows.

Lemma 46. *Let r be the root of tree T. Then there is a feasible block sequence for update flow network G iff $D_r[\pi] = 1$ for some X_r-permutation π.*

Proof. The lemma follows immediately from $V_r = V(G)$. ☐

In Sections _, _, _, and _, we show how to compute the table of a node in $O((k\ell)!\, k\ell n)$ time for each of the four node types leaf, introduce, forget, join, before we prove Theorem 45 in Section _.

☐ What about empty permutations, that is, if $X_x \subseteq \{t\}$?

☐ Argue somewhere that every vertex is contained in at most k blocks.

7.1 Leaf Nodes

Lemma 47. *Let $x \in V(T)$ be a leaf node and π be an X_x-permutation. Then $D_x[\pi] = 1$.*

Proof. Let $x \in V(T)$ be a leaf node. Then bag X_x contains a single vertex and $V_x = X_x$. Hence V_x does not induce any edges. Hence every X_x-permutation is V_x-feasible. Thus $D_x[\pi] = 1$ for every X_x-permutation π. □

Lemma 48. *Let $x \in V(T)$ be a leaf node. Then the table of x can be computed in time $O(k!\,)$.*

Proof. Let $x \in V(T)$ be a leaf node. Then bag X_x contains a single vertex. Hence, by Lemma 47, each of the at most $(k|X_x|)! = k!$ entries of table D_x can be computed in $O(1)$ time. Thus the entire table D_x can be computed in $O(k!\,)$ time. □

7.2 Introduce Nodes

Let $x \in V(T)$ be an introduce node with child y. Then $X_x \supseteq X_y$ and $V_x = X_x \cup V_y$.

Lemma 49. *Blocks induce subtrees.*

☐ Move.

Lemma 50. *$E(V_x) = E(X_x) \cup E(V_y)$.*

☐ Move.

Proof. □

Lemma 51. *Let $X \subseteq V(G)$ be a set of vertices, $Y \subseteq X$, π be an X-permutation, and π' be the Y-restriction of π. If π is X-feasible, then π' is Y-feasible.*

☐ Move?

Proof. Let $X \subseteq V(G)$ be a set of vertices, $Y \subseteq X$, π be an X-permutation, and π' be the Y-restriction of π. Suppose π is X-feasible. We show that π' is Y-feasible.

Let $e \in E(Y)$ and $i \in [|\pi'|]$. We first show the following claim.

Claim 52. *Let $P \in \mathcal{P}$. If $\alpha_P(e, B'_i) = \text{active}$, then $\alpha_P(e, B_{\pi(\pi'_i)}) = \text{active}$.*

Proof. Let $u, v \in V(G)$ be two vertices such that $e = (u, v)$ and $P \in \mathcal{P}$. Suppose $\alpha_P((u, v), B'_i) = \text{active}$. If $(u, v) \in E(P^u)$ and $b(u, P) \in B'_i$, then $b(u, P) \in B_{\pi(\pi'_i)}$ and hence $\alpha_P((u, v), B_{\pi(\pi'_i)}) = \text{active}$. Now suppose $(u, v) \in E(P^o)$ and $b(u, P) \notin B'_i$. If $b(u, P) \notin B_{\pi(\pi'_i)}$, then $\alpha_P((u, v), B_{\pi(\pi'_i)}) = \text{active}$. Now suppose $b(u, P) \in B_{\pi(\pi'_i)}$. Hence $\pi(b(u, P)) \leq \pi(\pi'_i)$. Moreover, since $(u, v) \in E(Y)$, $u \in Y$. Since π' is a Y-permutation, $b(u, P) \in S(\pi')$. But since $b(u, P) \notin B'_i$, $\pi'(b(u, P)) > i = \pi'(\pi'_i)$. Hence π and π' are not consistent which contradicts that π' is a restriction of π. ■

Since π is X-feasible and $e \in E(Y) \subseteq E(X)$, by Claim 52, we have

$$\sum_{P \in \mathcal{P} : \alpha_P(e, B'_i) = \text{active}} d_P \leq \sum_{P \in \mathcal{P} : \alpha_P(e, B_{\pi(\pi'_i)}) = \text{active}} d_P \leq c(e).$$

□

□ Point out that $\pi(\pi'_i)$ is defined, as π is an extension of π'.

□ Define B'_i.

□ Argue that $B'_i \subseteq B_{\pi(\pi'_i)}$.

Lemma 53. *Let $X, Y \subseteq V(G)$ be two sets of vertices, π be an X-feasible permutation, and π' be a Y-feasible permutation. If π and π' are consistent, then every union of π and π' is congestion free w.r.t. $E(X) \cup E(Y)$.*

□ Move?

Proof. Let $X, Y \subseteq V(G)$ be two sets of vertices, π be an X-feasible permutation, and π' be a Y-feasible permutation. Let $\tilde{\pi}$ be any union of π and π'. Suppose π and π' are consistent. We show that $\tilde{\pi}$ is congestion free w.r.t. $E(X) \cup E(Y)$.

Let $e \in E(X) \cup E(Y)$ and $i \in [|\tilde{\pi}|]$. We only consider the case $e \in E(X)$; the case $e \in E(Y)$ is similar. Suppose $e \in E(X)$. Let $j \in \{0, \dots, |\pi|\}$ such that $(\pi_1, \dots, \pi_j)$ is the X-restriction of $(\tilde{\pi}_1, \dots, \tilde{\pi}_i)$. We first show the following claim.

Claim 54. *Let $P \in \mathcal{P}$. If $\alpha_P(e, \tilde{B}_i) = \text{active}$, then $\alpha_P(e, B_j) = \text{active}$.*

Proof. Let $u, v \in V(G)$ be two vertices such that $e = (u, v)$ and $P \in \mathcal{P}$. Suppose $\alpha_P((u, v), \tilde{B}_i) = \text{active}$. If $(u, v) \in E(P^o)$ and $b(u, P) \notin \tilde{B}_i$, then $b(u, P) \notin B_j$ and hence $\alpha_P((u, v), B_j) = \text{active}$. Now suppose $(u, v) \in E(P^u)$ and $b(u, P) \in \tilde{B}_i$. If $b(u, P) \in B_j$, then $\alpha_P((u, v), B_j) = \text{active}$. Now suppose $b(u, P) \notin B_j$. Since $(u, v) \in E(X)$, $u \in X$. Since π is an X-permutation, $b(u, P) \in S(\pi)$. Hence $\pi(b(u, P)) > j = \pi(\pi_j)$. But since $b(u, P) \in \tilde{B}_i$ and by the choice of index j, $\tilde{\pi}(b(u, P)) \leq \tilde{\pi}(\pi_j)$. Hence π and $\tilde{\pi}$ are not consistent which contradicts that $\tilde{\pi}$ is a union of π and π'. ■

Since π is X-feasible and $e \in E(X)$, by Claim 54, we have

$$\sum_{P \in \mathcal{P} : \alpha_P(e, \tilde{B}_i) = \text{active}} d_P \leq \sum_{P \in \mathcal{P} : \alpha_P(e, B_j) = \text{active}} d_P \leq c(e).$$

□

□ Define $\tilde{B}_i$.

□ Argue that $B_j \subseteq \tilde{B}_i$.

Lemma 55 ([cite:@amiri2016congestionfree; Lemma 4.3)**.** *] Let $X \subseteq V(G)$ be a set of vertices and π be an X-permutation. Whether π is X-feasible can be determined in time $O(|\pi| \cdot |G|)$.*

- ☐ Do we have to change the running time of the algorithm to use $|G|$ instead of n?

Lemma 56. *Let $x \in V(T)$ be an introduce node with child y, π be an X_x-permutation, and π' be the X_y-restriction of π. Then $D_x[\pi] = 1$ iff π is X_x-feasible and $D_y[\pi'] = 1$.*

Proof. Let $x \in V(T)$ be an introduce node with child y. Then $X_x \supseteq X_y$ and $V_x = X_x \cup V_y$. Moreover, let π_{X_x} be an X_x-permutation and π_{X_y} be the X_y-restriction of π_{X_x}.

If part. Suppose $D_x[\pi_{X_x}] = 1$. Then let π_{V_x} be a V_x-feasible extension of π_{X_x}. Hence, by Lemma 51, π_{X_x} is X_x-feasible. To show $D_y[\pi_{X_y}] = 1$, consider the V_y-restriction π_{V_y} of π_{V_x}. Since $V_y \subseteq V_x$ and π_{V_x} is V_x-feasible, by Lemma 51, π_{V_y} is V_y-feasible. Moreover, since $X_y \subseteq X_x \subseteq V_x$, by Lemma 32 1, π_{V_x} is a V_x-extension of π_{X_y}. Since $X_y \subseteq V_y \subseteq V_x$, by Lemma 32 2, π_{V_y} is a V_y-extension of π_{X_y}. Thus $D_y[\pi_{X_y}] = 1$.

Only-if part. Suppose π_{X_x} is X_x-feasible and $D_y[\pi_{X_y}] = 1$. Then let π_{V_y} be a V_y-feasible extension of π_{X_y}. We show that π_{X_x} and π_{V_y} are consistent. Then, by Lemma 53, every union of π_{X_x} and π_{V_y} is congestion free w.r.t. $E(X_x) \cup E(V_y)$, which, by Lemma 50, equals $E(V_x)$. Thus, since every union of π_{X_x} and π_{V_y} is an extension of π_{X_x} to blocks induced by $X_x \cup V_y = V_x$, $D_x[\pi_{X_x}] = 1$.

Suppose not. Then obtain blocks b, b' such that $\pi_{X_x}(b) < \pi_{X_x}(b')$ and $\pi_{V_y}(b) > \pi_{V_y}(b')$. By Lemma 49, both b and b' are induced by X_y. If $\pi_{X_y}(b) < \pi_{X_y}(b')$, then π_{V_y} is not an extension of π_{X_y}. If $\pi_{X_y}(b) > \pi_{X_y}(b')$, then π_{X_y} is not a restriction of π_{X_x}. ☐

Lemma 57. *Let $x \in V(T)$ be an introduce node with child y. Given the table of y, the table of x can be computed in time $O((k|X_x|)!\, k|X_x|n)$.*

Proof. Let $x \in V(T)$ be an introduce node with child y. Lemma 56 shows how to compute table D_x given D_y. We can compute any restriction of a permutation π in $O(|\pi|)$ time. Moreover, by Lemma 55, we can determine whether a permutation π is feasible in $O(|\pi|n)$ time. Hence each of the at most $(k|X_x|)!$ entries of table D_x can be computed in $O(k|X_x|n)$ time. Thus the entire table D_x can be computed in $O((k|X_x|)!\, k|X_x|n)$ time. ☐

- ☐ Do we have to argue that any restriction of permutation π can be computed in $O(|\pi|)$ time?

7.3 Forget nodes

Lemma 58. *Let $x \in V(T)$ be a forget node with child y and π be an X_x-permutation. Then $D_x[\pi] = 1$ iff $D_y[\pi'] = 1$ for some X_y-extension π' of π.*

Proof. Let $x \in V(T)$ be a forget node with child y. Then $X_x \subseteq X_y$ and hence $V_x = V_y$. Moreover, let π_{X_x} be an X_x-permutation. The lemma follows from the following observation: Since $X_x \subseteq X_y \subseteq V_y = V_x$, every V_x-extension of π_{X_x} is the V_y-extension of some X_y-extension of π_{X_x}.

If part. Suppose $D_x[\pi_{X_x}] = 1$. Then let π_{V_x} be a V_x-feasible extension of π_{X_x}. Consider the X_y-restriction π_{X_y} of π_{V_x}. Since $X_x \subseteq X_y \subseteq V_x$, by Lemma 32 2, π_{X_y} is an X_y-extension of π_{X_x}. Moreover, since $V_x = V_y$, π_{V_x} is a V_y-feasible extension of π_{X_y} and thus $D_y[\pi_{X_y}] = 1$.

Only-if part. Suppose $D_y[\pi_{X_y}] = 1$ for some X_y-extension π_{X_y} of π_{X_x}. Then let π_{V_y} be a V_y-feasible extension of π_{X_y}. Since $X_x \subseteq X_y \subseteq V_y$, by Lemma 32 1, π_{V_y} is a V_y-extension of π_{X_x}. Moreover, since $V_x = V_y$, π_{V_y} is a V_x-feasible extension of π_{X_x} and thus $D_x[\pi_{X_x}] = 1$. □

```
foreach $X_x$-permutation $\pi$ do
    $D_x[\pi] \leftarrow 0$
end
foreach $X_y$-permutation $\pi'$ do
    {
end
if $D_y[\pi'] = 1$ then
    {
end
Compute the $X_x$-restriction $\pi$ of $\pi'$\;$D_x[\pi] \leftarrow 1$\;} }
```

□ Fix rendering.

Lemma 59. *Let $x \in V(T)$ be a forget node with child y. Given the table of y, the table of x can be computed in time $O((k|X_y|)!\, k|X_y|)$.*

Proof. Let $x \in V(T)$ be a forget node with child y. Then $X_x \subseteq X_y$. Lemma 58 shows how to compute table D_x given D_y. Notice, however, that an X_x-permutation may have many X_y-extensions, whereas an X_y-permutation has a unique X_x-restriction. Hence we use the fact that permutation π' is an X_y-extension of X_x-permutation π iff π is the X_x-restriction of π' and compute table D_x using Algorithm 10.

We show that, given forget node x and table D_y, the output D_x of Algorithm 10 satisfies Lemma 58. Let π be an X_x-permutation. If $D_x[\pi] = 0$, then $D_x[\pi]$

was not set to 1 in Line 10 and hence π is not the X_x-restriction of any X_y-permutation π' with $D_y[\pi'] = 1$, which is the case iff there is no X_y-extension π' of π with $D_y[\pi'] = 1$. If, on the other hand, $D_x[\pi] = 1$, then there is an X_y-permutation π' with $D_y[\pi'] = 1$ such that π is the X_x-restriction of π', which is the case iff there is an X_y-extension π' of π with $D_y[\pi'] = 1$.

Since we can compute any restriction of a permutation π' in $O(|\pi'|)$ time, Algorithm 10 computes the table for forget node x given the table for its child y in $O((k|X_x|)! + (k|X_y|)!\, k|X_y|) = O((k|X_y|)!\, k|X_y|)$ time. □

7.4 Join nodes

Lemma 60. $E(V_x) = E(V_{y_1}) \cup E(V_{y_2})$.

□ Move.

Proof. □

Lemma 61. *Let $x \in V(T)$ be a join node with children y_1 and y_2 and π be an X_x-permutation. Then $D_x[\pi] = 1$ iff both $D_{y_1}[\pi] = 1$ and $D_{y_2}[\pi] = 1$.*

Proof. Let $x \in V(T)$ be a join node with children y_1 and y_2. Then $X_{y_1} = X_x = X_{y_2}$ and hence $V_x = X_x \cup V_{y_1} \cup V_{y_2} = V_{y_1} \cup V_{y_2}$. Moreover, let π_{X_x} be an X_x-permutation.

If part. Suppose $D_x[\pi_{X_x}] = 1$. Then let π_{V_x} be a V_x-feasible extension of π_{X_x}. Consider the V_{y_1}-restriction $\pi_{V_{y_1}}$ of π_{V_x}. Since $V_{y_1} \subseteq V_x$ and π_{V_x} is V_x-feasible, by Lemma 51, $\pi_{V_{y_1}}$ is V_{y_1}-feasible. Moreover, since $X_x = X_{y_1} \subseteq V_{y_1} \subseteq V_x$, by Lemma 32 2, $\pi_{V_{y_1}}$ is a V_{y_1}-extension of π_{X_x}. Thus $D_{y_1}[\pi_{X_x}] = 1$. Analogously, $D_{y_2}[\pi_{X_x}] = 1$.

Only-if part. Suppose both $D_{y_1}[\pi_{X_x}] = 1$ and $D_{y_2}[\pi_{X_x}] = 1$. Then let $\pi_{V_{y_1}}$ ($\pi_{V_{y_2}}$) be a V_{y_1}-feasible (V_{y_2}-feasible) extension of π_{X_x}. We show that $\pi_{V_{y_1}}$ and $\pi_{V_{y_2}}$ are consistent. Then, by Lemma 53, every union of $\pi_{V_{y_1}}$ and $\pi_{V_{y_2}}$ is congestion free w.r.t. $E(V_{y_1}) \cup E(V_{y_2})$, which, by Lemma 60 equals $E(V_x)$. Moreover, since every union $\pi_{V_{y_1} \cup V_{y_2}}$ of $\pi_{V_{y_1}}$ and $\pi_{V_{y_2}}$ is an extension of $\pi_{V_{y_1}}$ to blocks induced by $V_{y_1} \cup V_{y_2} = V_x$, $\pi_{V_{y_1} \cup V_{y_2}}$ is a V_x-extension of $\pi_{V_{y_1}}$. Hence, since $X_x = X_{y_1} \subseteq V_{y_1} \subseteq V_x$, by Lemma 32 1, $\pi_{V_{y_1} \cup V_{y_2}}$ is a V_x-extension of π_{X_x}. Thus $D_x[\pi_{X_x}] = 1$.

Suppose $\pi_{V_{y_1}}$ and $\pi_{V_{y_2}}$ are not consistent. Then obtain blocks b, b' such that $\pi_{V_{y_1}}(b) < \pi_{V_{y_1}}(b')$ and $\pi_{V_{y_2}}(b) > \pi_{V_{y_2}}(b')$. By Lemma 49, both b and b' are induced by X_x. If $\pi_{X_x}(b) < \pi_{X_x}(b')$, then $\pi_{V_{y_2}}$ is not an extension of π_{X_x}. If $\pi_{X_x}(b) > \pi_{X_x}(b')$, then $\pi_{V_{y_1}}$ is not an extension of π_{X_x}. □

Lemma 62. *Let $x \in V(T)$ be a join node with children y_1 and y_2. Given the tables of y_1, y_2, the table of x can be computed in time $O((k|X_x|)!\,)$.*

Proof. Let $x \in V(T)$ be a join node with children y_1 and y_2. Lemma 61 shows how to compute table D_x given D_{y_1} and D_{y_2}. Hence each of the at most $(k|X_x|)!$ entries of table D_x can be computed in $O(1)$ time. Thus the entire table D_x can be computed in $O((k|X_x|)!\,)$ time. □

7.5 Putting everything together

A *postorder tree walk* is an ordering of the nodes of a tree such that every node is later in the ordering than any of its descendants. There are simple and well-known algorithms to find a postorder tree walk of a given tree in linear time.

In our algorithm, we compute tables for nodes in postorder, that is, first we determine a postorder tree walk for tree T, and then we compute for each node $x \in V(T)$ table D_x in this postorder. In this way, all tables for the children of x are already computed before we compute D_x, and we thus can use the methods described above to compute the tables.

Suppose r is the root of tree T. After all other tables have been computed, we compute table D_r. Given this table, we can decide if there is a feasible block sequence for update flow network G.

☐ Edit! This is basically a copy-paste from (Bodlaender, Hans L. and Koster, Arie MCA, 2008).

Lemma 63. *There is a feasible block sequence for update flow network G iff $D_r[\pi] = 1$ for some X_r-permutation π.*

Proof. The lemma follows immediately from $V_r = V(G)$. □

[[thm:algorithm-bounded-treewidth-decision.]] Since tree decomposition $(T, \mathcal{X})$ has width $\ell - 1$, for every node $x \in V(T)$, we can compute table D_x in $O((k\ell)!\,k\ell n)$ time. Since $(T, \mathcal{X})$ has $O(n)$ nodes, we can compute the tables for all nodes in $O((k\ell)!\,k\ell n^2)$ time. Moreover, by Lemma 63, given the table for the root, we can decide if there is a feasible block sequence for update flow network G in $O((k\ell)!\,)$ time. Thus the algorithm decides if there is a feasible block sequence for G in $O((k\ell)!\,k\ell n^2)$ time. □

☐ Should we add lemmas for the running times?

Chapter 8

Search Problem