Parameterized Algorithms

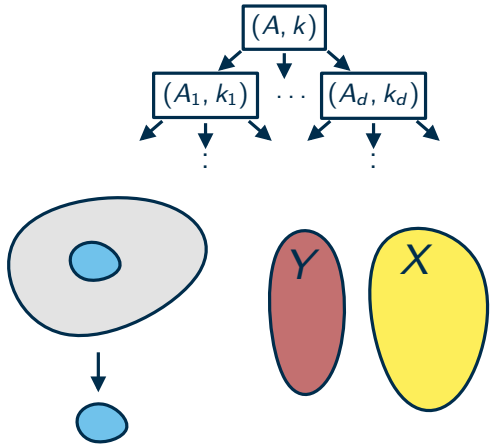
Above Lower Bound, Duality, and Lenstra

Thomas Bläsius

Content

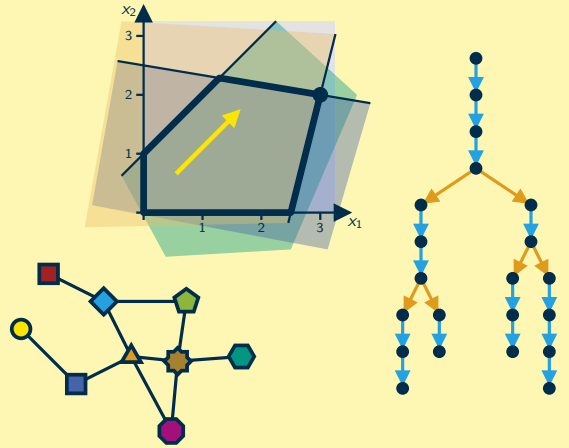
Basic toolbox

- bounded search trees
- kernelization
- iterative compression



Extended toolbox

- linear programs
- branch-and-reduce
- color coding

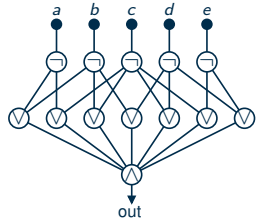


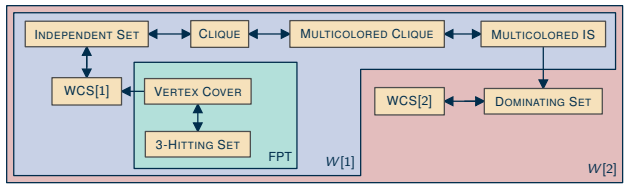
Tree width

- dynamic programming
- chordal and planar graphs
- Courcelle's theorem

Lower bonds

- kernel lower bounds
- parameterized reductions
- circuits and the W-hierarchy
- ETH and SETH



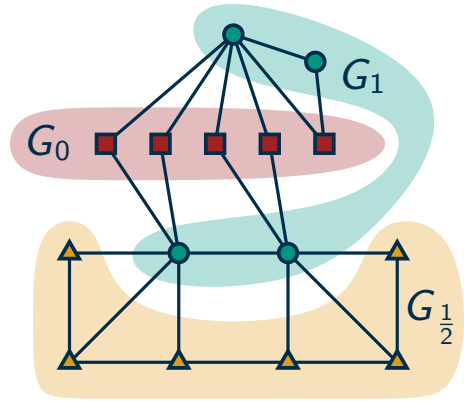


Recap: Kernelization via LP relaxation

minimize: $\sum_{v \in V} x_v$

such that: $x_u + x_v \geq 1$ for $uv \in E$

$0 \leq x_v \leq 1$ for $v \in V$



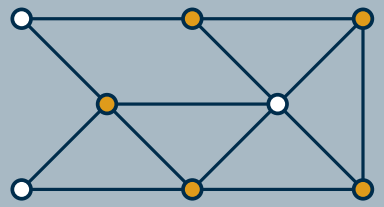
Reduction rule

- solve the LP relaxation $\rightarrow (x_v)_{v \in V}$
- if $\sum_{v \in V} x_v > k$: constant size `no`-instance
- otherwise: delete $V_0 \cup V_1$ and reduce k by $|V_1| \rightarrow G_{\frac{1}{2}}$ is the kernel

Lemma: The reduction rule is safe.

Recap: Bounded search tree

Problem: VERTEX COVER
Given a graph $G = (V, E)$ and a parameter k. Does G have a vertex cover of size k?

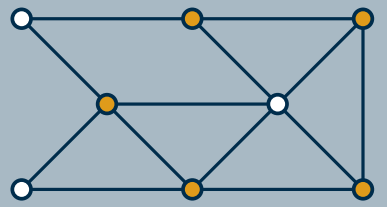


(vertex set $V' \subseteq V$ with $e \cap V' \neq \emptyset$ for all $e \in E$)

Is there a vertex cover of size at most $k = 3$?

Recap: Bounded search tree

Problem: VERTEX COVER
Given a graph $G = (V, E)$ and a parameter k. Does G have a vertex cover of size k?



(vertex set $V' \subseteq V$ with $e \cap V' \neq \emptyset$ for all $e \in E$)

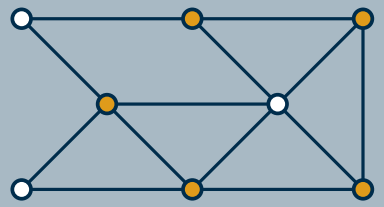
Is there a vertex cover of size at most $k = 3$?

Simple branching algorithm

- start: $V' = \emptyset \rightarrow$ no edge covered

Recap: Bounded search tree

Problem: VERTEX COVER
Given a graph $G = (V, E)$ and a parameter k. Does G have a vertex cover of size k?



(vertex set $V' \subseteq V$ with $e \cap V' \neq \emptyset$ for all $e \in E$)

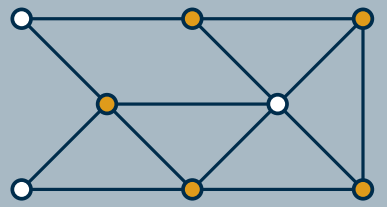
Is there a vertex cover of size at most $k = 3$?

Simple branching algorithm

- start: $V' = \emptyset \rightarrow$ no edge covered
- step: branch on uncovered edge

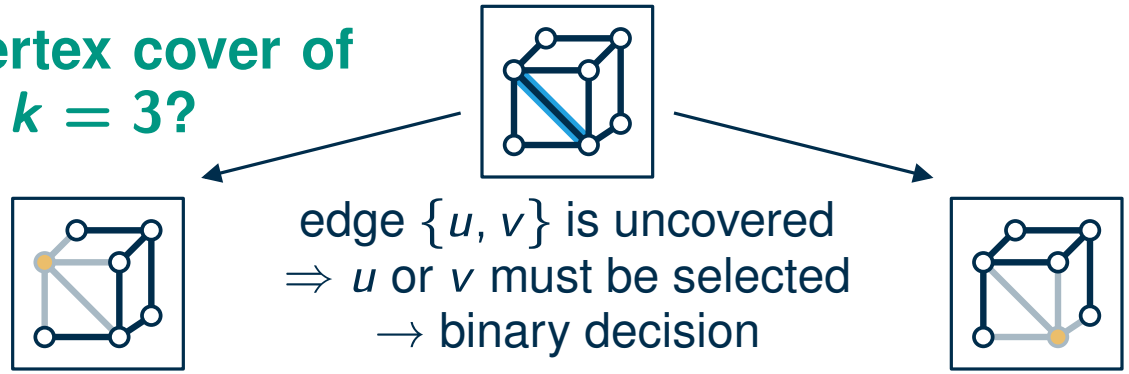
Recap: Bounded search tree

Problem: VERTEX COVER
Given a graph $G = (V, E)$ and a parameter k. Does G have a vertex cover of size k?



(vertex set $V' \subseteq V$ with $e \cap V' \neq \emptyset$ for all $e \in E$)

Is there a vertex cover of size at most $k = 3$?



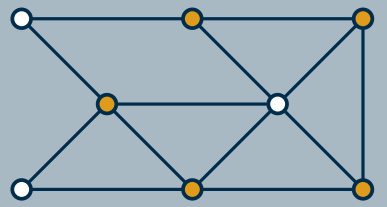
edge $\{u, v\}$ is uncovered
$\Rightarrow$ u or v must be selected
$\rightarrow$ binary decision

Simple branching algorithm

- start: $V' = \emptyset \rightarrow$ no edge covered
- step: branch on uncovered edge

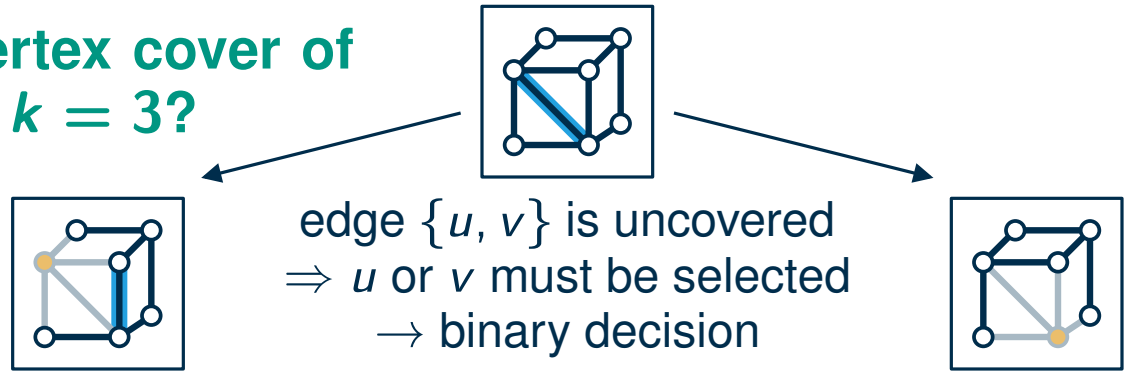
Recap: Bounded search tree

Problem: VERTEX COVER
Given a graph $G = (V, E)$ and a parameter k. Does G have a vertex cover of size k?



(vertex set $V' \subseteq V$ with $e \cap V' \neq \emptyset$ for all $e \in E$)

Is there a vertex cover of size at most $k = 3$?



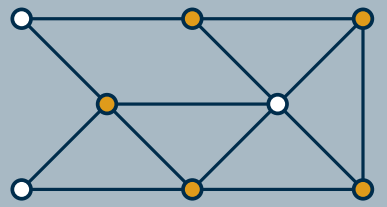
edge $\{u, v\}$ is uncovered
$\Rightarrow$ u or v must be selected
$\rightarrow$ binary decision

Simple branching algorithm

- start: $V' = \emptyset \rightarrow$ no edge covered
- step: branch on uncovered edge

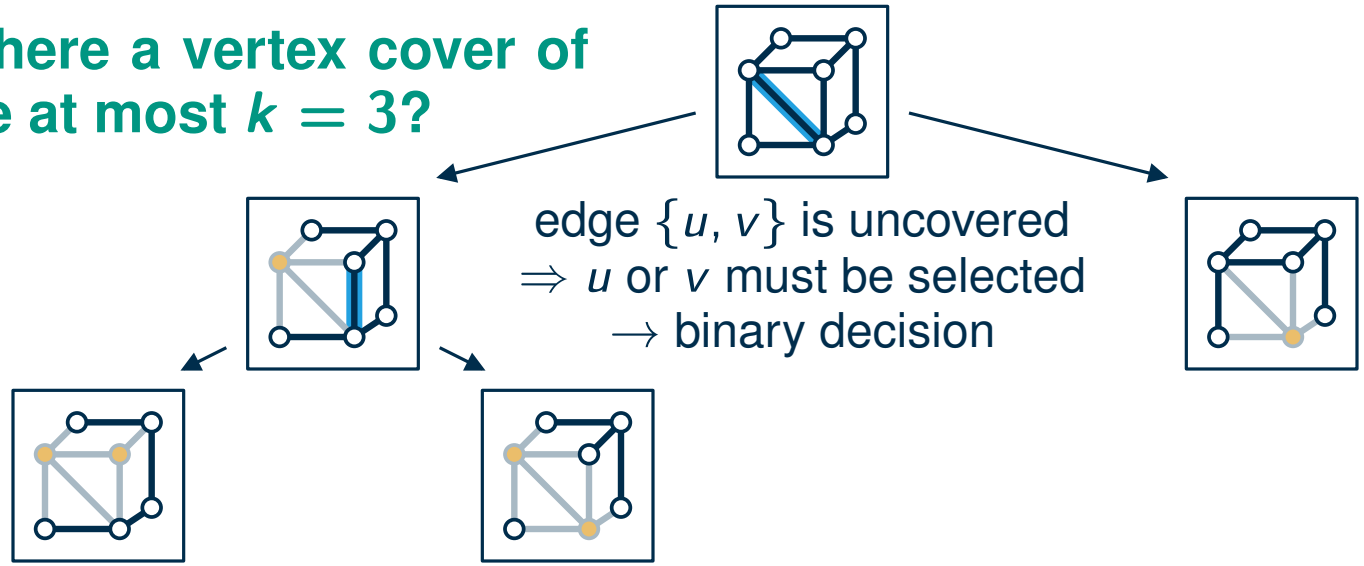
Recap: Bounded search tree

Problem: VERTEX COVER
Given a graph $G = (V, E)$ and a parameter k. Does G have a vertex cover of size k?



(vertex set $V' \subseteq V$ with $e \cap V' \neq \emptyset$ for all $e \in E$)

Is there a vertex cover of size at most $k = 3$?



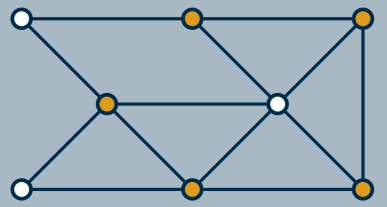
edge $\{u, v\}$ is uncovered
$\Rightarrow$ u or v must be selected
$\rightarrow$ binary decision

Simple branching algorithm

- start: $V' = \emptyset \rightarrow$ no edge covered
- step: branch on uncovered edge

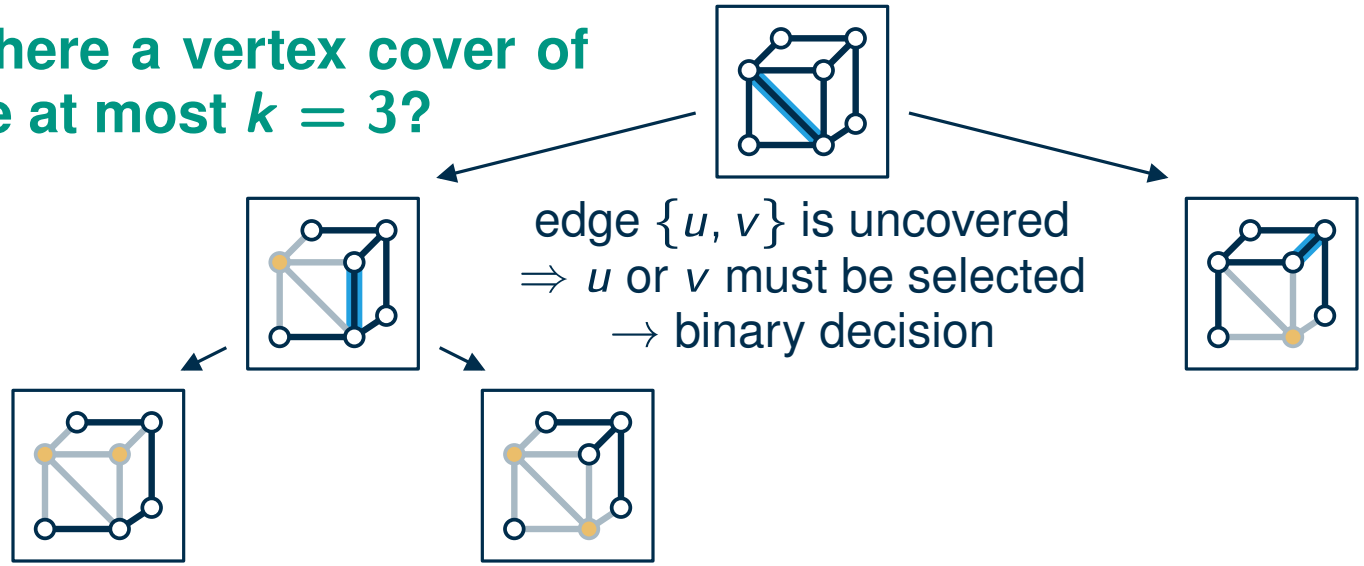
Recap: Bounded search tree

Problem: VERTEX COVER
Given a graph $G = (V, E)$ and a parameter k. Does G have a vertex cover of size k?



(vertex set $V' \subseteq V$ with $e \cap V' \neq \emptyset$ for all $e \in E$)

Is there a vertex cover of size at most $k = 3$?



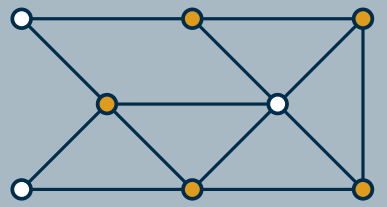
edge $\{u, v\}$ is uncovered
$\Rightarrow$ u or v must be selected
$\rightarrow$ binary decision

Simple branching algorithm

- start: $V' = \emptyset \rightarrow$ no edge covered
- step: branch on uncovered edge

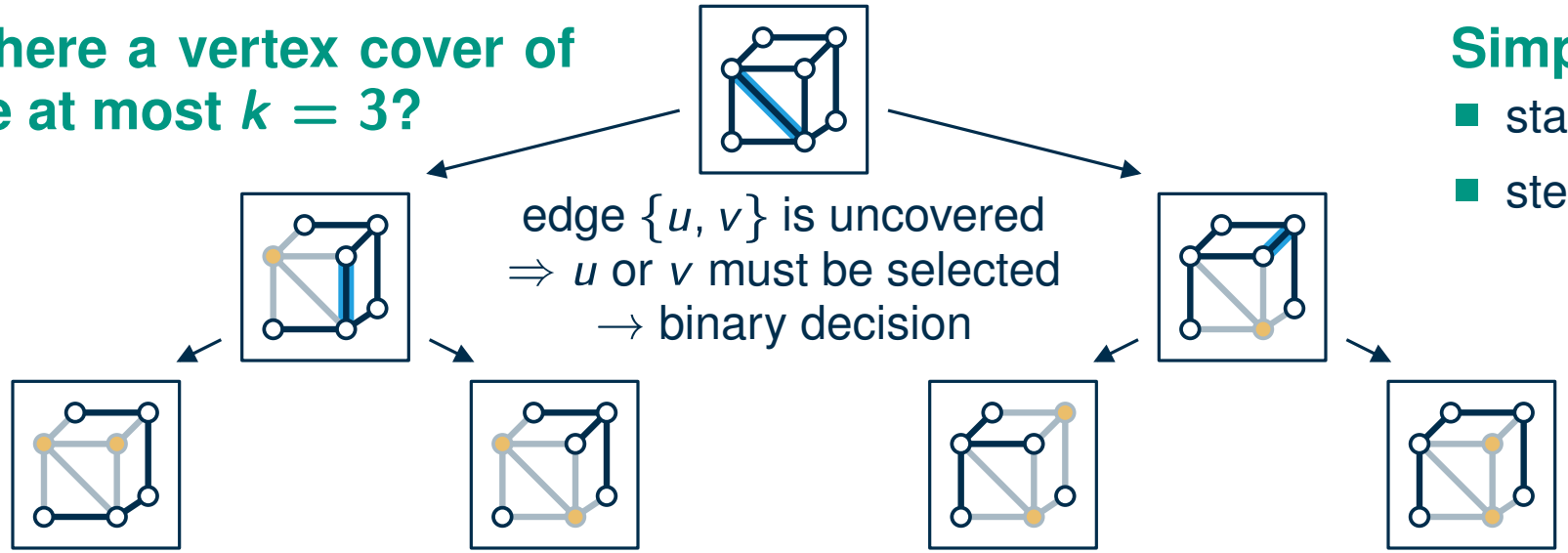
Recap: Bounded search tree

Problem: VERTEX COVER
Given a graph $G = (V, E)$ and a parameter k. Does G have a vertex cover of size k?



(vertex set $V' \subseteq V$ with $e \cap V' \neq \emptyset$ for all $e \in E$)

Is there a vertex cover of size at most $k = 3$?



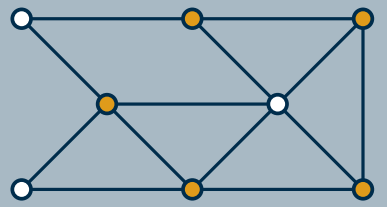
edge $\{u, v\}$ is uncovered
$\Rightarrow$ u or v must be selected
$\rightarrow$ binary decision

Simple branching algorithm

- start: $V' = \emptyset \rightarrow$ no edge covered
- step: branch on uncovered edge

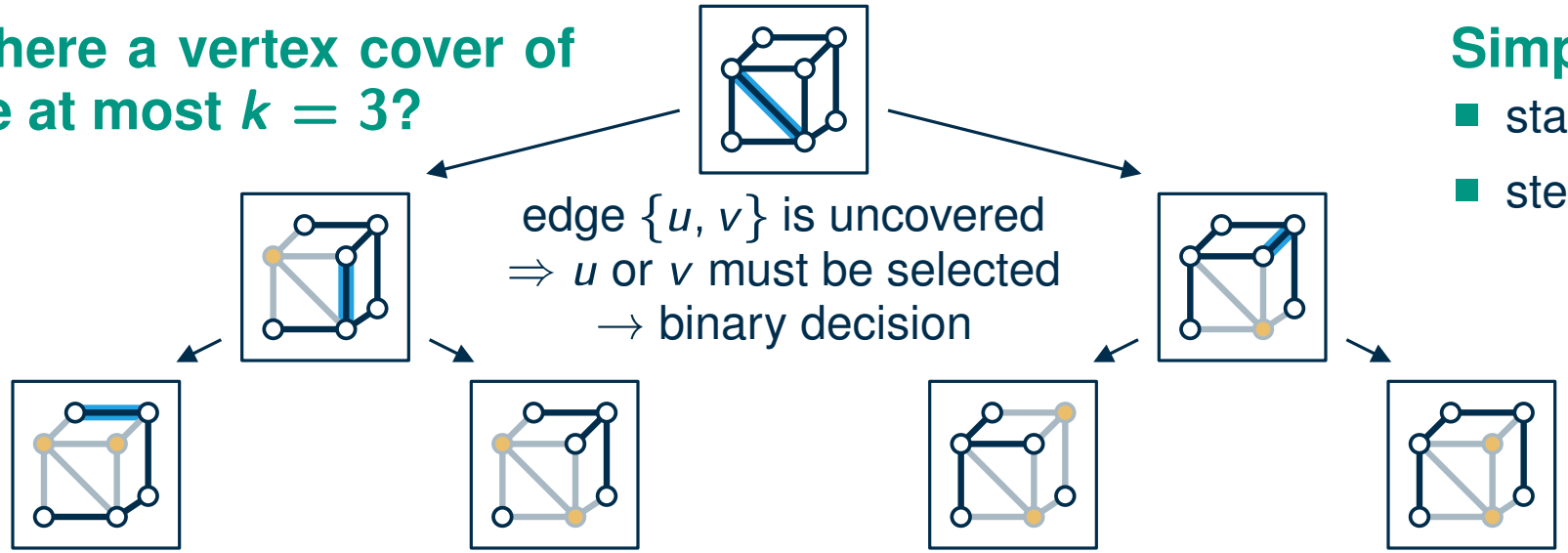
Recap: Bounded search tree

Problem: VERTEX COVER
Given a graph $G = (V, E)$ and a parameter k. Does G have a vertex cover of size k?



(vertex set $V' \subseteq V$ with $e \cap V' \neq \emptyset$ for all $e \in E$)

Is there a vertex cover of size at most $k = 3$?



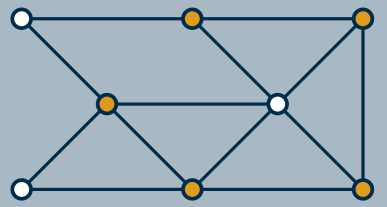
edge $\{u, v\}$ is uncovered
$\Rightarrow$ u or v must be selected
$\rightarrow$ binary decision

Simple branching algorithm

- start: $V' = \emptyset \rightarrow$ no edge covered
- step: branch on uncovered edge

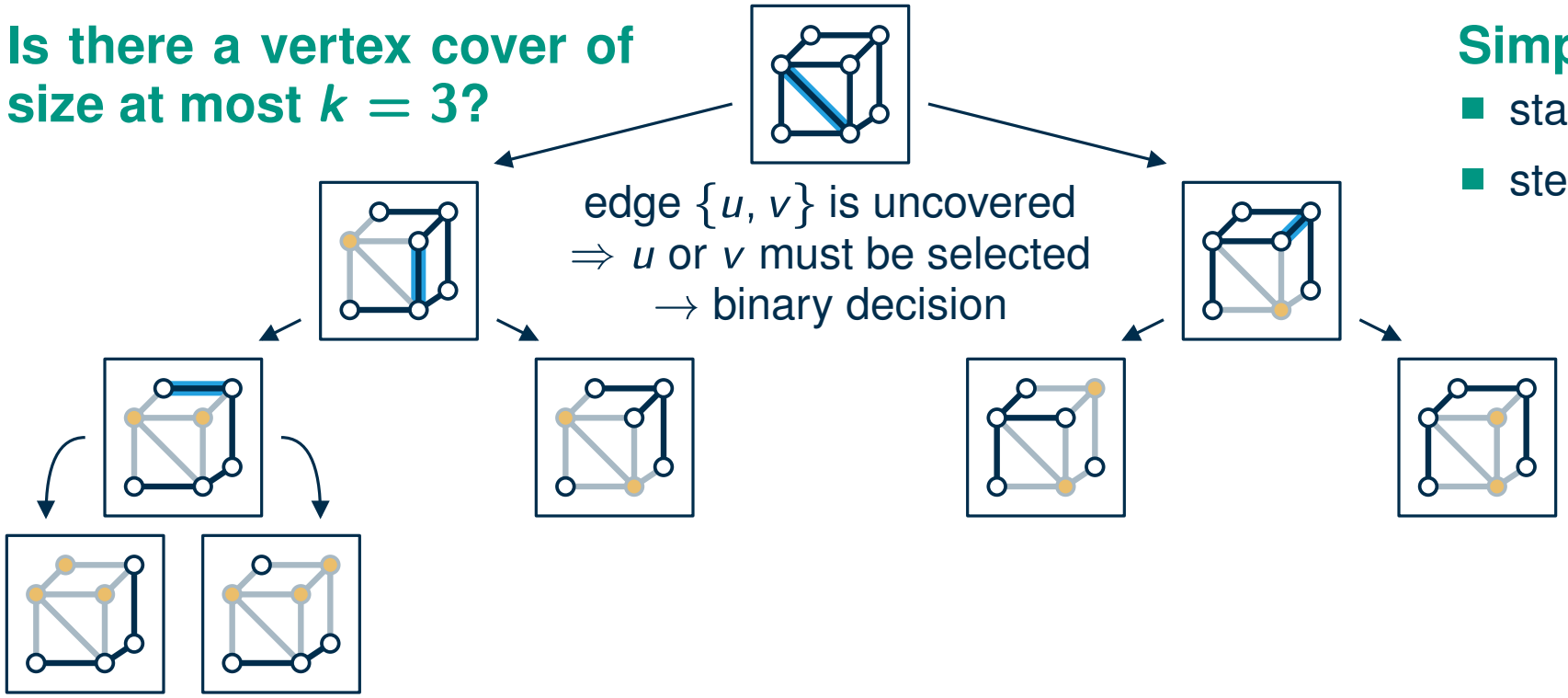
Recap: Bounded search tree

Problem: VERTEX COVER
Given a graph $G = (V, E)$ and a parameter k. Does G have a vertex cover of size k?



(vertex set $V' \subseteq V$ with $e \cap V' \neq \emptyset$ for all $e \in E$)

Is there a vertex cover of size at most $k = 3$?



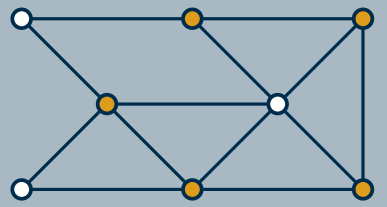
edge $\{u, v\}$ is uncovered
$\Rightarrow$ u or v must be selected
$\rightarrow$ binary decision

Simple branching algorithm

- start: $V' = \emptyset \rightarrow$ no edge covered
- step: branch on uncovered edge

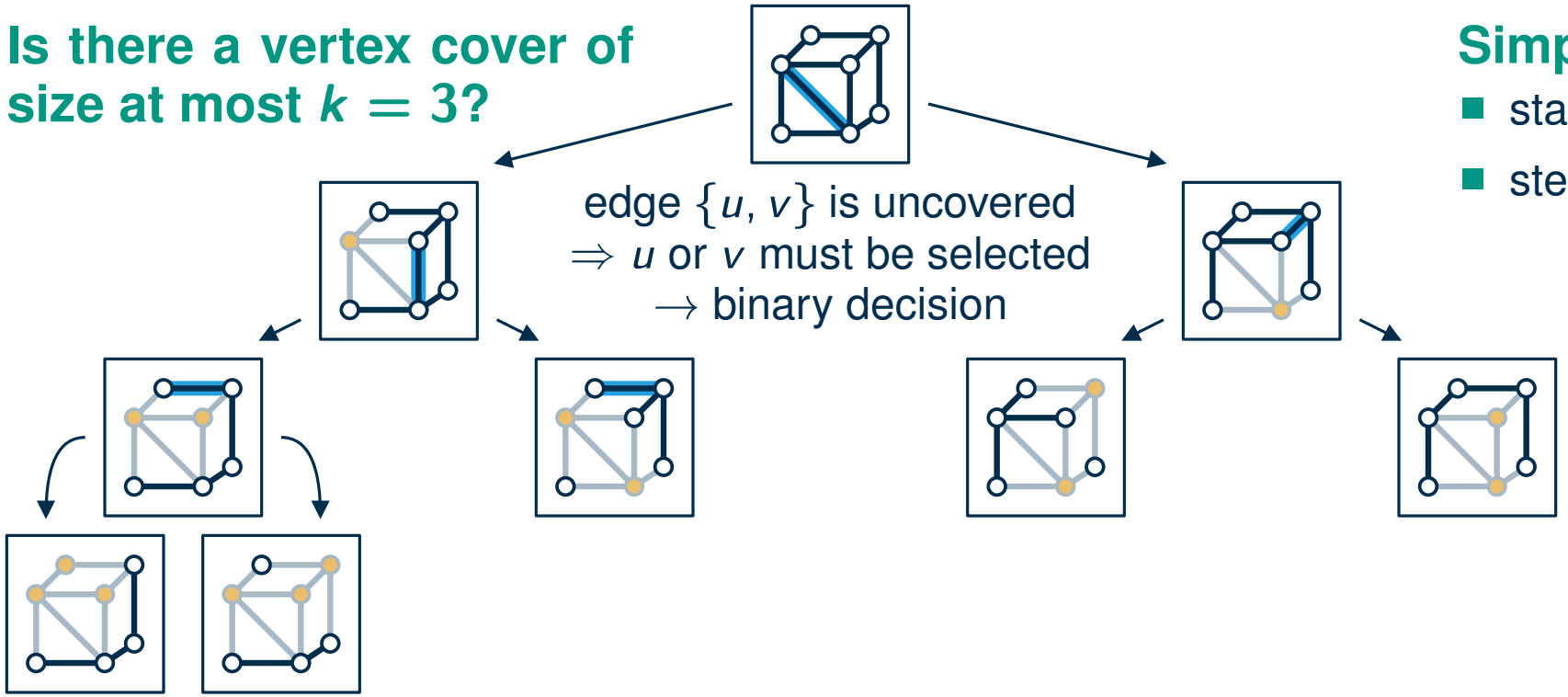
Recap: Bounded search tree

Problem: VERTEX COVER
Given a graph $G = (V, E)$ and a parameter k. Does G have a vertex cover of size k?



(vertex set $V' \subseteq V$ with $e \cap V' \neq \emptyset$ for all $e \in E$)

Is there a vertex cover of size at most $k = 3$?



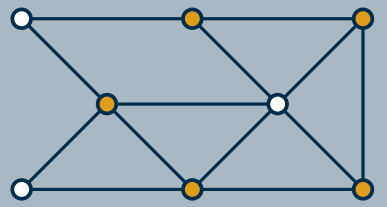
edge $\{u, v\}$ is uncovered
$\Rightarrow$ u or v must be selected
$\rightarrow$ binary decision

Simple branching algorithm

- start: $V' = \emptyset \rightarrow$ no edge covered
- step: branch on uncovered edge

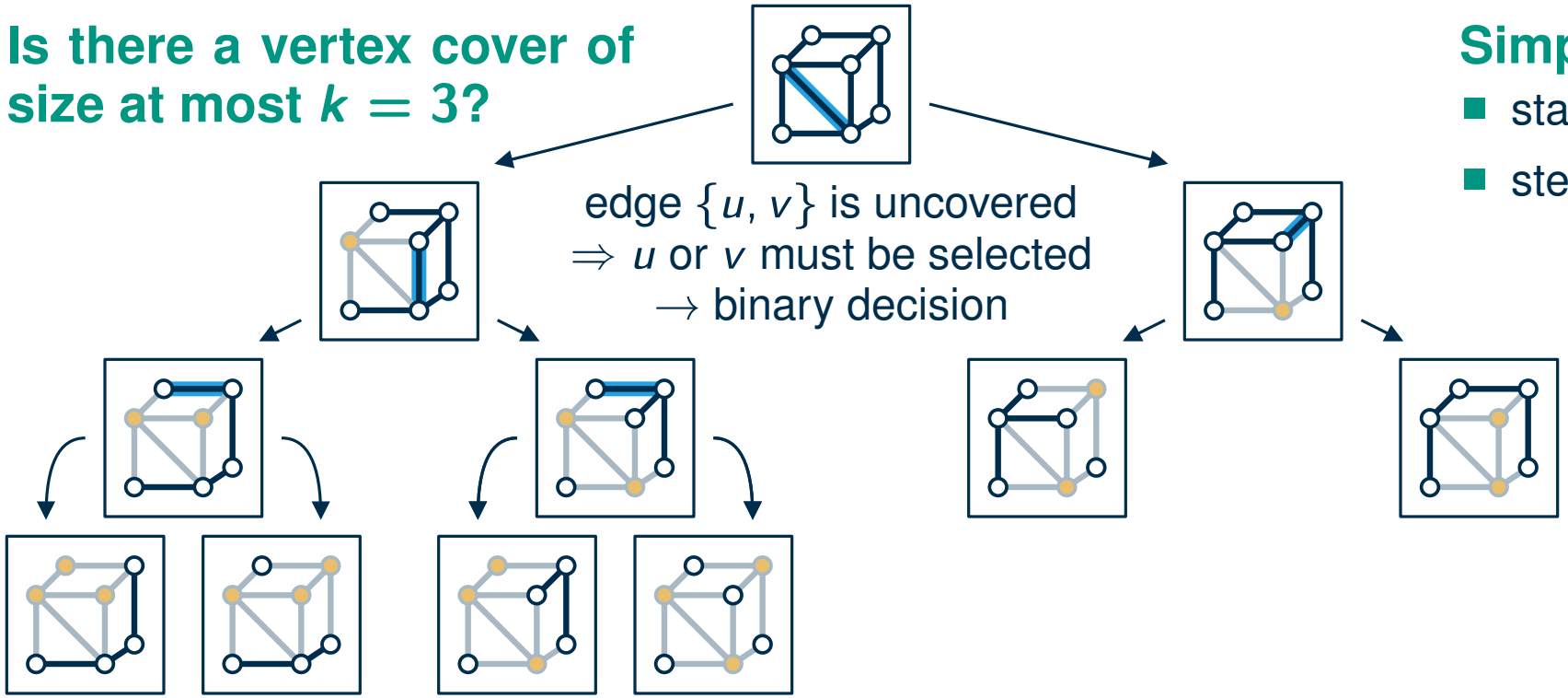
Recap: Bounded search tree

Problem: VERTEX COVER
Given a graph $G = (V, E)$ and a parameter k. Does G have a vertex cover of size k?



(vertex set $V' \subseteq V$ with $e \cap V' \neq \emptyset$ for all $e \in E$)

Is there a vertex cover of size at most $k = 3$?



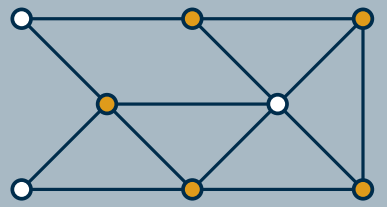
edge $\{u, v\}$ is uncovered
$\Rightarrow$ u or v must be selected
$\rightarrow$ binary decision

Simple branching algorithm

- start: $V' = \emptyset \rightarrow$ no edge covered
- step: branch on uncovered edge

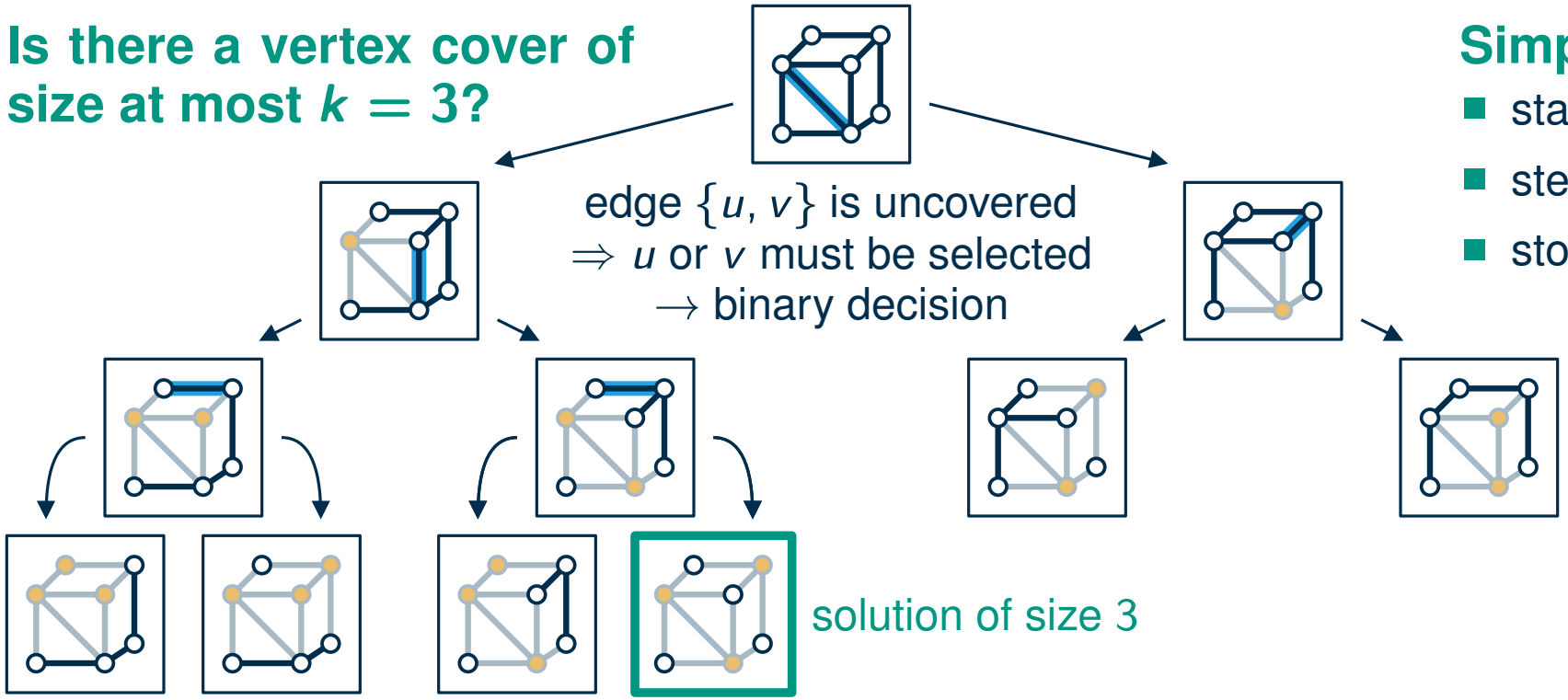
Recap: Bounded search tree

Problem: VERTEX COVER
Given a graph $G = (V, E)$ and a parameter k. Does G have a vertex cover of size k?



(vertex set $V' \subseteq V$ with $e \cap V' \neq \emptyset$ for all $e \in E$)

Is there a vertex cover of size at most $k = 3$?



edge $\{u, v\}$ is uncovered
$\Rightarrow$ u or v must be selected
$\rightarrow$ binary decision

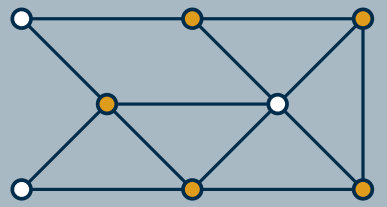
solution of size 3

Simple branching algorithm

- start: $V' = \emptyset \rightarrow$ no edge covered
- step: branch on uncovered edge
- stop if: all edges covered or already k vertices selected

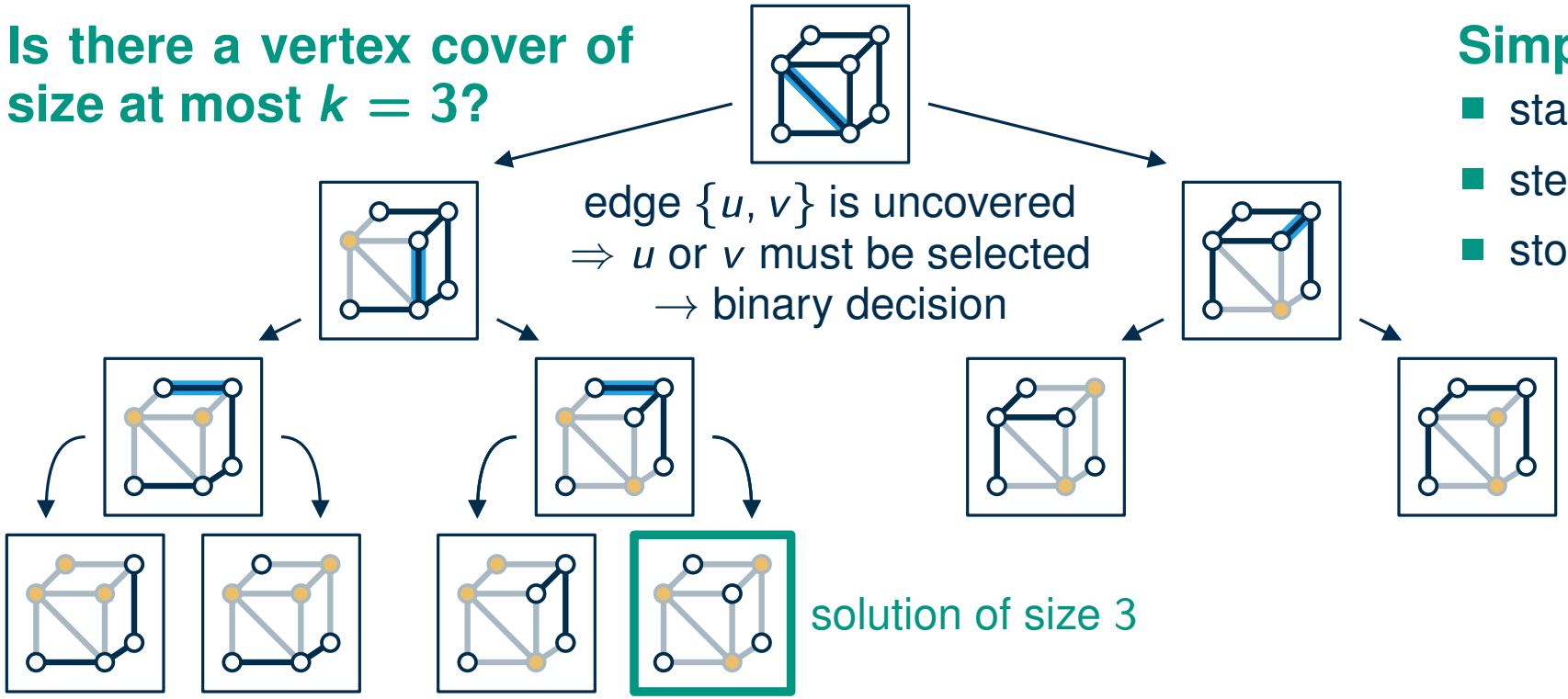
Recap: Bounded search tree

Problem: VERTEX COVER
Given a graph $G = (V, E)$ and a parameter k. Does G have a vertex cover of size k?



(vertex set $V' \subseteq V$ with $e \cap V' \neq \emptyset$ for all $e \in E$)

Is there a vertex cover of size at most $k = 3$?



edge $\{u, v\}$ is uncovered
$\Rightarrow$ u or v must be selected
$\rightarrow$ binary decision

solution of size 3

Simple branching algorithm

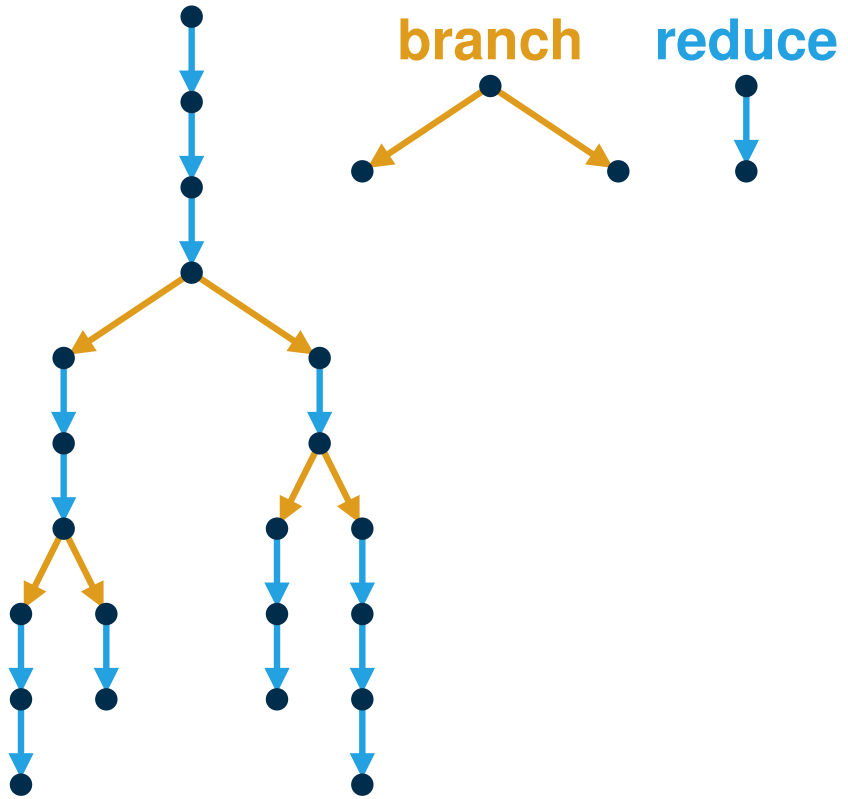
- start: $V' = \emptyset \rightarrow$ no edge covered
- step: branch on uncovered edge
- stop if: all edges covered or already k vertices selected

running time: $O(2^k m)$

Kernelization and search trees

Branch-and-reduce

- apply reduction rules whenever possible
- no rule applicable: branch once



Kernelization and search trees

Branch-and-reduce

- apply reduction rules whenever possible
- no rule applicable: branch once
- for VERTEX COVER: highly efficient in practice

Instance					B&R
Name	$\|V\|$	$\|E\|$	LP	VC	T
Social networks:					
ca-GrQc	5242	14,484	2412.5	2783	0.1
ca-HepTh	9877	25,973	4553.0	4981	0.2
ca-CondMat	23,133	93,439	11,156.5	13,521	0.1
wiki-Vote	7115	100,762	2249.0	2249	0.0
ca-HepPh	12,008	118,489	5722.5	7012	0.1
email-Enron	36,692	183,831	12,559.5	14,437	0.6
ca-AstroPh	18,772	198,050	9218.5	12,012	0.1
email-EuAll	265,214	364,481	18,315.5	18,316	0.1
soc-Epinions1	75,879	405,740	22,080.0	22,280	1.1
soc-Slashdot0811	77,360	469,180	23,936.0	24,046	0.2
soc-Slashdot0902	82,168	504,230	25,657.5	25,770	0.2
dblp-2010	300,647	807,700	138,634.5	166,234	2.0
youtube-links	1,138,499	2,990,443	276,684.0	278,125	1.0
dblp-2011	933,258	3,353,618	427,093.5	498,969	4.8
wiki-Talk	2,394,385	4,659,565	56,111.5	56,163	1.3
petster-friendships-cat	149,700	5,448,197	40,975.0	41,103	3.0
petster-friendships-dog	426,820	8,543,549	149,419.5	150,117	5.8
youtube-u-growth	3,223,589	9,376,594	834,673.5	840,060	2.6
flickr-links	1,715,255	15,555,041	467,585.5	474,637	2.1
petster-carnivore	623,766	15,695,166	290,474.0	371,189	3.6
libimseti	220,970	17,233,144	93,378.0	93,676	1642.8
soc-pokec	1,632,803	22,301,964	781,066.0	–	–
flickr-growth	2,302,925	22,838,276	632,011.0	640,921	2.8
soc-LiveJournal1	4,847,571	42,851,237	2,110,273.5	2,215,668	11.5
hollywood-2009	1,107,243	56,375,711	553,223.0	890,039	26.8
hollywood-2011	1,985,306	114,492,816	992,238.0	1,657,357	50.5
orkut-links	3,072,441	117,185,083	1,519,101.5	–	–

Akibaa, Iwata: *Branch-and-reduce exponential/FPT algorithms in practice: A case study of vertex cover* (2016)

Kernelization and search trees

Branch-and-reduce

- apply reduction rules whenever possible
- no rule applicable: branch once
- for VERTEX COVER: highly efficient in practice

FPT perspective

- explains why this works well in practice

Instance					B&R
Name	$\|V\|$	$\|E\|$	LP	VC	T
Social networks:					
ca-GrQc	5242	14,484	2412.5	2783	0.1
ca-HepTh	9877	25,973	4553.0	4981	0.2
ca-CondMat	23,133	93,439	11,156.5	13,521	0.1
wiki-Vote	7115	100,762	2249.0	2249	0.0
ca-HepPh	12,008	118,489	5722.5	7012	0.1
email-Enron	36,692	183,831	12,559.5	14,437	0.6
ca-AstroPh	18,772	198,050	9218.5	12,012	0.1
email-EuAll	265,214	364,481	18,315.5	18,316	0.1
soc-Epinions1	75,879	405,740	22,080.0	22,280	1.1
soc-Slashdot0811	77,360	469,180	23,936.0	24,046	0.2
soc-Slashdot0902	82,168	504,230	25,657.5	25,770	0.2
dblp-2010	300,647	807,700	138,634.5	166,234	2.0
youtube-links	1,138,499	2,990,443	276,684.0	278,125	1.0
dblp-2011	933,258	3,353,618	427,093.5	498,969	4.8
wiki-Talk	2,394,385	4,659,565	56,111.5	56,163	1.3
petster-friendships-cat	149,700	5,448,197	40,975.0	41,103	3.0
petster-friendships-dog	426,820	8,543,549	149,419.5	150,117	5.8
youtube-u-growth	3,223,589	9,376,594	834,673.5	840,060	2.6
flickr-links	1,715,255	15,555,041	467,585.5	474,637	2.1
petster-carnivore	623,766	15,695,166	290,474.0	371,189	3.6
libimseti	220,970	17,233,144	93,378.0	93,676	1642.8
soc-pokec	1,632,803	22,301,964	781,066.0	–	–
flickr-growth	2,302,925	22,838,276	632,011.0	640,921	2.8
soc-LiveJournal1	4,847,571	42,851,237	2,110,273.5	2,215,668	11.5
hollywood-2009	1,107,243	56,375,711	553,223.0	890,039	26.8
hollywood-2011	1,985,306	114,492,816	992,238.0	1,657,357	50.5
orkut-links	3,072,441	117,185,083	1,519,101.5	–	–

Akibaa, Iwata: *Branch-and-reduce exponential/FPT algorithms in practice: A case study of vertex cover* (2016)

Kernelization and search trees

Branch-and-reduce

- apply reduction rules whenever possible
- no rule applicable: branch once
- for VERTEX COVER: highly efficient in practice

FPT perspective

- explains why this works well in practice

Really?

Instance					B&R
Name	$\|V\|$	$\|E\|$	LP	VC	T
Social networks:					
ca-GrQc	5242	14,484	2412.5	2783	0.1
ca-HepTh	9877	25,973	4553.0	4981	0.2
ca-CondMat	23,133	93,439	11,156.5	13,521	0.1
wiki-Vote	7115	100,762	2249.0	2249	0.0
ca-HepPh	12,008	118,489	5722.5	7012	0.1
email-Enron	36,692	183,831	12,559.5	14,437	0.6
ca-AstroPh	18,772	198,050	9218.5	12,012	0.1
email-EuAll	265,214	364,481	18,315.5	18,316	0.1
soc-Epinions1	75,879	405,740	22,080.0	22,280	1.1
soc-Slashdot0811	77,360	469,180	23,936.0	24,046	0.2
soc-Slashdot0902	82,168	504,230	25,657.5	25,770	0.2
dblp-2010	300,647	807,700	138,634.5	166,234	2.0
youtube-links	1,138,499	2,990,443	276,684.0	278,125	1.0
dblp-2011	933,258	3,353,618	427,093.5	498,969	4.8
wiki-Talk	2,394,385	4,659,565	56,111.5	56,163	1.3
petster-friendships-cat	149,700	5,448,197	40,975.0	41,103	3.0
petster-friendships-dog	426,820	8,543,549	149,419.5	150,117	5.8
youtube-u-growth	3,223,589	9,376,594	834,673.5	840,060	2.6
flickr-links	1,715,255	15,555,041	467,585.5	474,637	2.1
petster-carnivore	623,766	15,695,166	290,474.0	371,189	3.6
libimseti	220,970	17,233,144	93,378.0	93,676	1642.8
soc-pokec	1,632,803	22,301,964	781,066.0	–	–
flickr-growth	2,302,925	22,838,276	632,011.0	640,921	2.8
soc-LiveJournal1	4,847,571	42,851,237	2,110,273.5	2,215,668	11.5
hollywood-2009	1,107,243	56,375,711	553,223.0	890,039	26.8
hollywood-2011	1,985,306	114,492,816	992,238.0	1,657,357	50.5
orkut-links	3,072,441	117,185,083	1,519,101.5	–	–

Akibaa, Iwata: *Branch-and-reduce exponential/FPT algorithms in practice: A case study of vertex cover* (2016)

Kernelization and search trees

Branch-and-reduce

- apply reduction rules whenever possible
- no rule applicable: branch once
- for VERTEX COVER: highly efficient in practice

FPT perspective

- explains why this works well in practice
- $O(2^k n)$ does not explain much if $n = 1.1\,\text{M}$ and $k = 0.9\,\text{M}$

Really?

Instance					B&R
Name	$\|V\|$	$\|E\|$	LP	VC	T
Social networks:					
ca-GrQc	5242	14,484	2412.5	2783	0.1
ca-HepTh	9877	25,973	4553.0	4981	0.2
ca-CondMat	23,133	93,439	11,156.5	13,521	0.1
wiki-Vote	7115	100,762	2249.0	2249	0.0
ca-HepPh	12,008	118,489	5722.5	7012	0.1
email-Enron	36,692	183,831	12,559.5	14,437	0.6
ca-AstroPh	18,772	198,050	9218.5	12,012	0.1
email-EuAll	265,214	364,481	18,315.5	18,316	0.1
soc-Epinions1	75,879	405,740	22,080.0	22,280	1.1
soc-Slashdot0811	77,360	469,180	23,936.0	24,046	0.2
soc-Slashdot0902	82,168	504,230	25,657.5	25,770	0.2
dblp-2010	300,647	807,700	138,634.5	166,234	2.0
youtube-links	1,138,499	2,990,443	276,684.0	278,125	1.0
dblp-2011	933,258	3,353,618	427,093.5	498,969	4.8
wiki-Talk	2,394,385	4,659,565	56,111.5	56,163	1.3
petster-friendships-cat	149,700	5,448,197	40,975.0	41,103	3.0
petster-friendships-dog	426,820	8,543,549	149,419.5	150,117	5.8
youtube-u-growth	3,223,589	9,376,594	834,673.5	840,060	2.6
flickr-links	1,715,255	15,555,041	467,585.5	474,637	2.1
petster-carnivore	623,766	15,695,166	290,474.0	371,189	3.6
libimseti	220,970	17,233,144	93,378.0	93,676	1642.8
soc-pokec	1,632,803	22,301,964	781,066.0	–	–
flickr-growth	2,302,925	22,838,276	632,011.0	640,921	2.8
soc-LiveJournal1	4,847,571	42,851,237	2,110,273.5	2,215,668	11.5
hollywood-2009	1,107,243	56,375,711	553,223.0	890,039	26.8
hollywood-2011	1,985,306	114,492,816	992,238.0	1,657,357	50.5
orkut-links	3,072,441	117,185,083	1,519,101.5	–	–

Akibaa, Iwata: *Branch-and-reduce exponential/FPT algorithms in practice: A case study of vertex cover* (2016)

Kernelization and search trees

Branch-and-reduce

- apply reduction rules whenever possible
- no rule applicable: branch once
- for VERTEX COVER: highly efficient in practice

FPT perspective

- explains why this works well in practice
- $O(2^k n)$ does not explain much if $n = 1.1\,\text{M}$ and $k = 0.9\,\text{M}$
- solution size is not a good parameter here

Really?

Instance					B&R
Name	$\|V\|$	$\|E\|$	LP	VC	T
Social networks:					
ca-GrQc	5242	14,484	2412.5	2783	0.1
ca-HepTh	9877	25,973	4553.0	4981	0.2
ca-CondMat	23,133	93,439	11,156.5	13,521	0.1
wiki-Vote	7115	100,762	2249.0	2249	0.0
ca-HepPh	12,008	118,489	5722.5	7012	0.1
email-Enron	36,692	183,831	12,559.5	14,437	0.6
ca-AstroPh	18,772	198,050	9218.5	12,012	0.1
email-EuAll	265,214	364,481	18,315.5	18,316	0.1
soc-Epinions1	75,879	405,740	22,080.0	22,280	1.1
soc-Slashdot0811	77,360	469,180	23,936.0	24,046	0.2
soc-Slashdot0902	82,168	504,230	25,657.5	25,770	0.2
dblp-2010	300,647	807,700	138,634.5	166,234	2.0
youtube-links	1,138,499	2,990,443	276,684.0	278,125	1.0
dblp-2011	933,258	3,353,618	427,093.5	498,969	4.8
wiki-Talk	2,394,385	4,659,565	56,111.5	56,163	1.3
petster-friendships-cat	149,700	5,448,197	40,975.0	41,103	3.0
petster-friendships-dog	426,820	8,543,549	149,419.5	150,117	5.8
youtube-u-growth	3,223,589	9,376,594	834,673.5	840,060	2.6
flickr-links	1,715,255	15,555,041	467,585.5	474,637	2.1
petster-carnivore	623,766	15,695,166	290,474.0	371,189	3.6
libimseti	220,970	17,233,144	93,378.0	93,676	1642.8
soc-pokec	1,632,803	22,301,964	781,066.0	–	–
flickr-growth	2,302,925	22,838,276	632,011.0	640,921	2.8
soc-LiveJournal1	4,847,571	42,851,237	2,110,273.5	2,215,668	11.5
hollywood-2009	1,107,243	56,375,711	553,223.0	890,039	26.8
hollywood-2011	1,985,306	114,492,816	992,238.0	1,657,357	50.5
orkut-links	3,072,441	117,185,083	1,519,101.5	–	–

Akibaa, Iwata: *Branch-and-reduce exponential/FPT algorithms in practice: A case study of vertex cover* (2016)

Kernelization and search trees

Branch-and-reduce

- apply reduction rules whenever possible
- no rule applicable: branch once
- for VERTEX COVER: highly efficient in practice

FPT perspective

- explains why this works well in practice
- $O(2^k n)$ does not explain much if $n = 1.1\,\text{M}$ and $k = 0.9\,\text{M}$
- solution size is not a good parameter here
- better: solution size $-$ value of the LP solution

Really?

Instance					B&R
Name	$\|V\|$	$\|E\|$	LP	VC	T
Social networks:					
ca-GrQc	5242	14,484	2412.5	2783	0.1
ca-HepTh	9877	25,973	4553.0	4981	0.2
ca-CondMat	23,133	93,439	11,156.5	13,521	0.1
wiki-Vote	7115	100,762	2249.0	2249	0.0
ca-HepPh	12,008	118,489	5722.5	7012	0.1
email-Enron	36,692	183,831	12,559.5	14,437	0.6
ca-AstroPh	18,772	198,050	9218.5	12,012	0.1
email-EuAll	265,214	364,481	18,315.5	18,316	0.1
soc-Epinions1	75,879	405,740	22,080.0	22,280	1.1
soc-Slashdot0811	77,360	469,180	23,936.0	24,046	0.2
soc-Slashdot0902	82,168	504,230	25,657.5	25,770	0.2
dblp-2010	300,647	807,700	138,634.5	166,234	2.0
youtube-links	1,138,499	2,990,443	276,684.0	278,125	1.0
dblp-2011	933,258	3,353,618	427,093.5	498,969	4.8
wiki-Talk	2,394,385	4,659,565	56,111.5	56,163	1.3
petster-friendships-cat	149,700	5,448,197	40,975.0	41,103	3.0
petster-friendships-dog	426,820	8,543,549	149,419.5	150,117	5.8
youtube-u-growth	3,223,589	9,376,594	834,673.5	840,060	2.6
flickr-links	1,715,255	15,555,041	467,585.5	474,637	2.1
petster-carnivore	623,766	15,695,166	290,474.0	371,189	3.6
libimseti	220,970	17,233,144	93,378.0	93,676	1642.8
soc-pokec	1,632,803	22,301,964	781,066.0	–	–
flickr-growth	2,302,925	22,838,276	632,011.0	640,921	2.8
soc-LiveJournal1	4,847,571	42,851,237	2,110,273.5	2,215,668	11.5
hollywood-2009	1,107,243	56,375,711	553,223.0	890,039	26.8
hollywood-2011	1,985,306	114,492,816	992,238.0	1,657,357	50.5
orkut-links	3,072,441	117,185,083	1,519,101.5	–	–

Akibaa, Iwata: *Branch-and-reduce exponential/FPT algorithms in practice: A case study of vertex cover* (2016)

Parameterization above lower bound

Solution size and lower bounds

- denote the size of a minimum vertex cover of G with $\mathrm{vc}(G)$
- let $\ell(G)$ be a lower bound on $\mathrm{vc}(G)$, i.e., we know $\mathrm{vc}(G) \geq \ell(G)$
- so far: parameter $\mathrm{vc}(G)$; now: parameter $\mathrm{vc}(G) - \ell(G)$

Parameterization above lower bound

Solution size and lower bounds

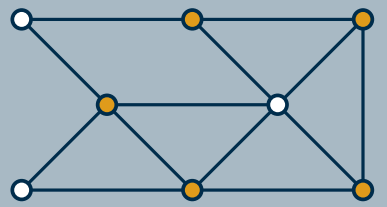
- denote the size of a minimum vertex cover of G with $\mathrm{vc}(G)$
- let $\ell(G)$ be a lower bound on $\mathrm{vc}(G)$, i.e., we know $\mathrm{vc}(G) \geq \ell(G)$
- so far: parameter $\mathrm{vc}(G)$; now: parameter $\mathrm{vc}(G) - \ell(G)$
- $\ell_{\mathrm{LP}}(G) =$ optimal solution of the LP-relaxation

Parameterization above lower bound

Solution size and lower bounds

- denote the size of a minimum vertex cover of G with $\mathrm{vc}(G)$
- let $\ell(G)$ be a lower bound on $\mathrm{vc}(G)$, i.e., we know $\mathrm{vc}(G) \geq \ell(G)$
- so far: parameter $\mathrm{vc}(G)$; now: parameter $\mathrm{vc}(G) - \ell(G)$
- $\ell_{\mathrm{LP}}(G) =$ optimal solution of the LP-relaxation

Problem: Vertex Cover above LP
Given a graph $G = (V, E)$ and a parameter k. Does G have a vertex cover of size $\ell_{\mathrm{LP}}(G) + k$?



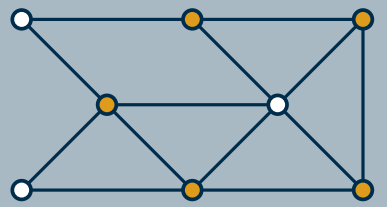
(vertex set $V' \subseteq V$ with $e \cap V' \neq \emptyset$ for all $e \in E$)

Parameterization above lower bound

Solution size and lower bounds

- denote the size of a minimum vertex cover of G with $\mathrm{vc}(G)$
- let $\ell(G)$ be a lower bound on $\mathrm{vc}(G)$, i.e., we know $\mathrm{vc}(G) \geq \ell(G)$
- so far: parameter $\mathrm{vc}(G)$; now: parameter $\mathrm{vc}(G) - \ell(G)$
- $\ell_{\mathrm{LP}}(G) =$ optimal solution of the LP-relaxation

Problem: Vertex Cover above LP
Given a graph $G = (V, E)$ and a parameter k. Does G have a vertex cover of size $\ell_{\mathrm{LP}}(G) + k$?



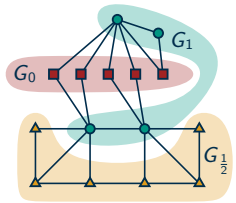
(vertex set $V' \subseteq V$ with $e \cap V' \neq \emptyset$ for all $e \in E$)

Plan for today

- FPT algorithm for Vertex Cover above LP
- note: $\mathrm{vc}(G)$ (previous parameter) can be much bigger than $\mathrm{vc}(G) - \ell_{\mathrm{LP}}(G)$ (new parameter)

Branch-and-reduce: How does k change?

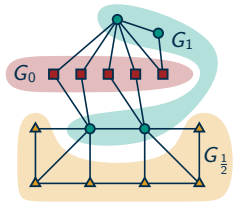
Reduce: if the LP relaxation has a solution with $G_{\frac{1}{2}} \subsetneq G$, reduce to $G_{\frac{1}{2}}$

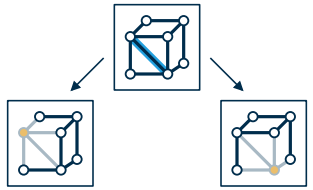


Branch-and-reduce: How does k change?

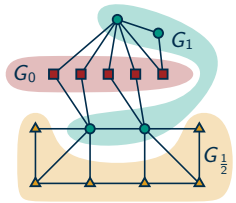
Reduce: if the LP relaxation has a solution with $G_{\frac{1}{2}} \subsetneq G$, reduce to $G_{\frac{1}{2}}$

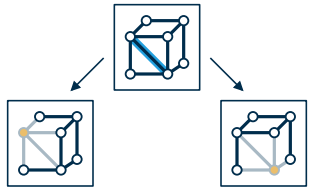
Branch: for an edge uv, consider the child instances $G - u$ and $G - v$





Branch-and-reduce: How does k change?



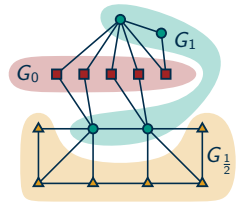


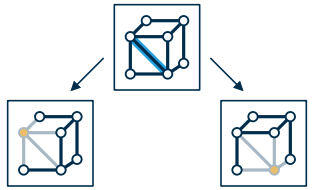
Reduce: if the LP relaxation has a solution with $G_{\frac{1}{2}} \subsetneq G$, reduce to $G_{\frac{1}{2}}$

Branch: for an edge uv, consider the child instances $G - u$ and $G - v$

How to adjust the parameter?

Branch-and-reduce: How does k change?

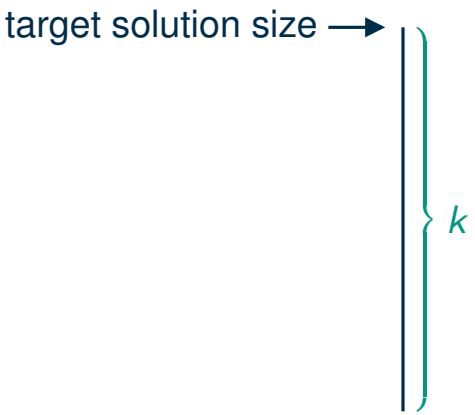




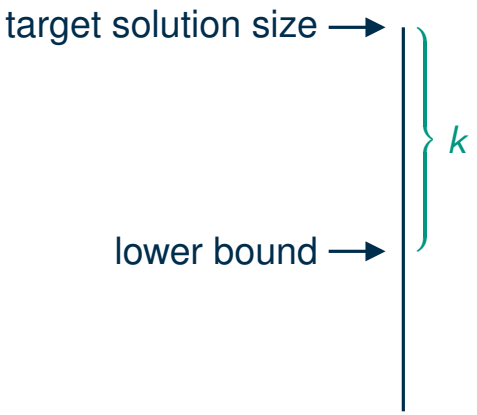
Reduce: if the LP relaxation has a solution with $G_{\frac{1}{2}} \subsetneq G$, reduce to $G_{\frac{1}{2}}$

Branch: for an edge uv, consider the child instances $G - u$ and $G - v$

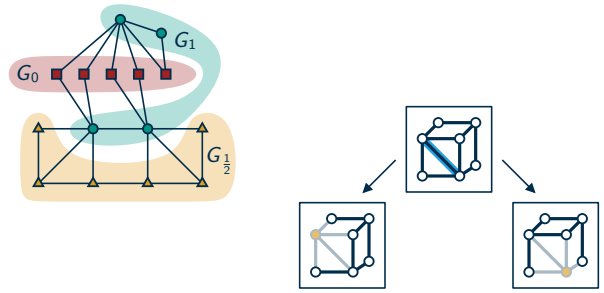
Problem: VERTEX COVER
Given a graph $G = (V, E)$ and a parameter k.
Does G have a vertex cover of size k?



Problem: VERTEX COVER ABOVE LP
Given a graph $G = (V, E)$ and a parameter k.
Does G have a vertex cover of size $\ell_{\text{LP}}(G)+k$?



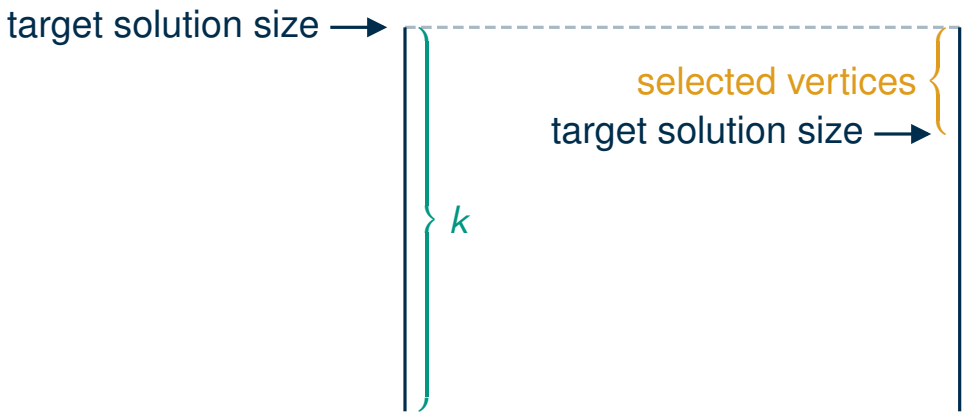
Branch-and-reduce: How does k change?



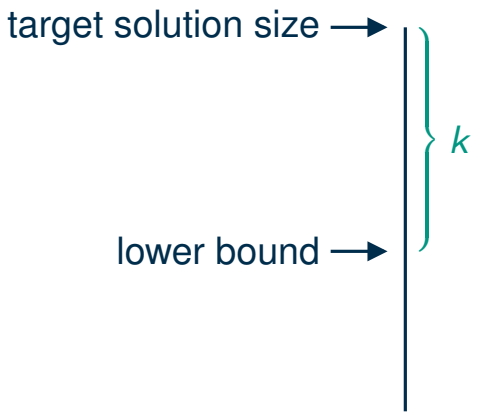
Reduce: if the LP relaxation has a solution with $G_{\frac{1}{2}} \subsetneq G$, reduce to $G_{\frac{1}{2}}$

Branch: for an edge uv, consider the child instances $G - u$ and $G - v$

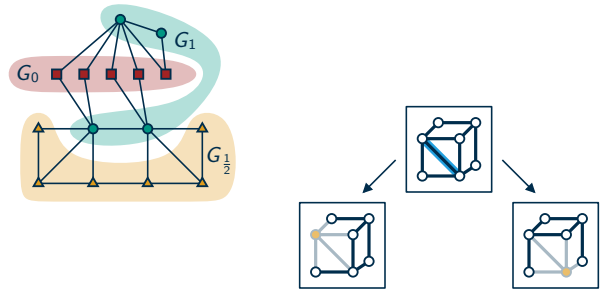
Problem: VERTEX COVER
Given a graph $G = (V, E)$ and a parameter k.
Does G have a vertex cover of size k?



Problem: VERTEX COVER ABOVE LP
Given a graph $G = (V, E)$ and a parameter k.
Does G have a vertex cover of size $\ell_{\text{LP}}(G)+k$?



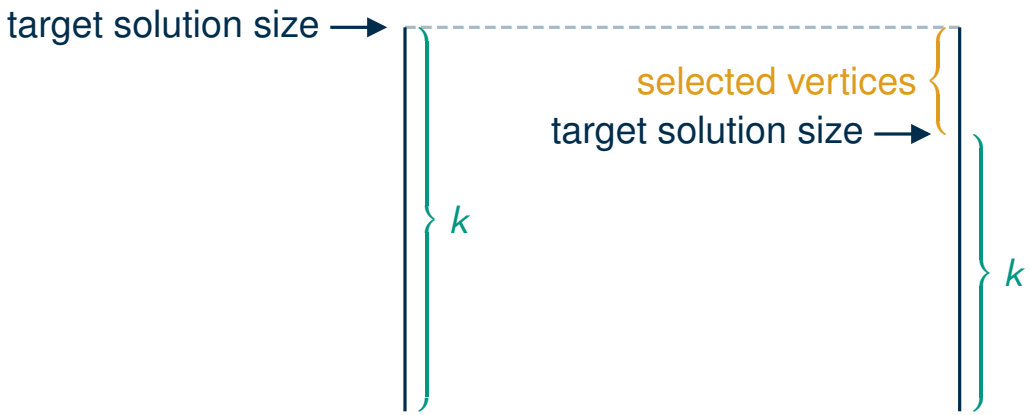
Branch-and-reduce: How does k change?



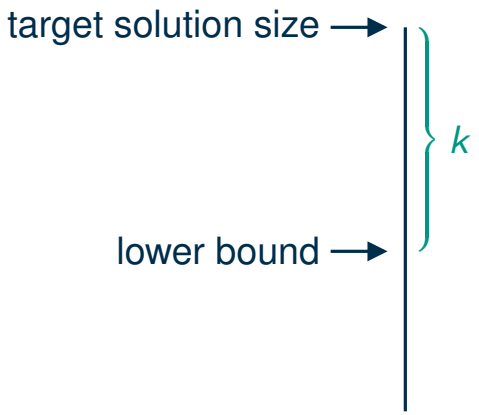
Reduce: if the LP relaxation has a solution with $G_{\frac{1}{2}} \subsetneq G$, reduce to $G_{\frac{1}{2}}$

Branch: for an edge uv, consider the child instances $G - u$ and $G - v$

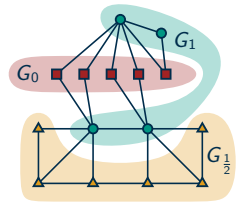
Problem: VERTEX COVER
Given a graph $G = (V, E)$ and a parameter k.
Does G have a vertex cover of size k?

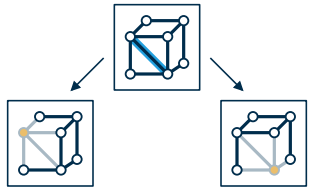


Problem: VERTEX COVER ABOVE LP
Given a graph $G = (V, E)$ and a parameter k.
Does G have a vertex cover of size $\ell_{\text{LP}}(G)+k$?



Branch-and-reduce: How does k change?

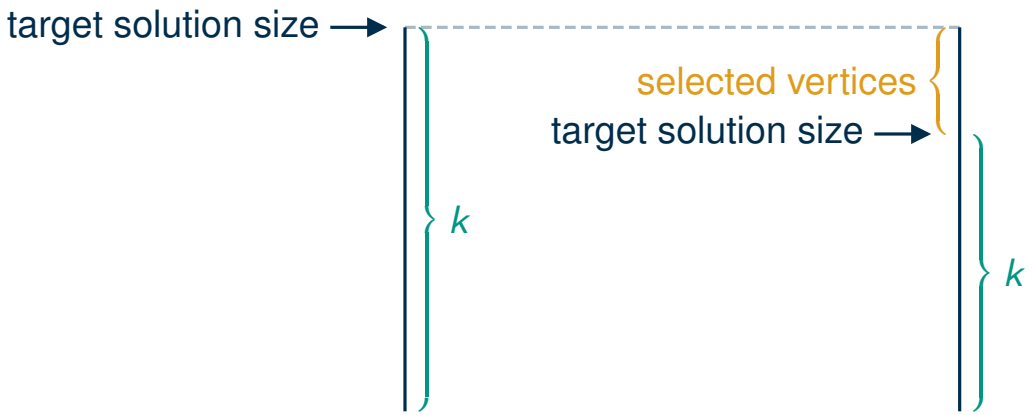




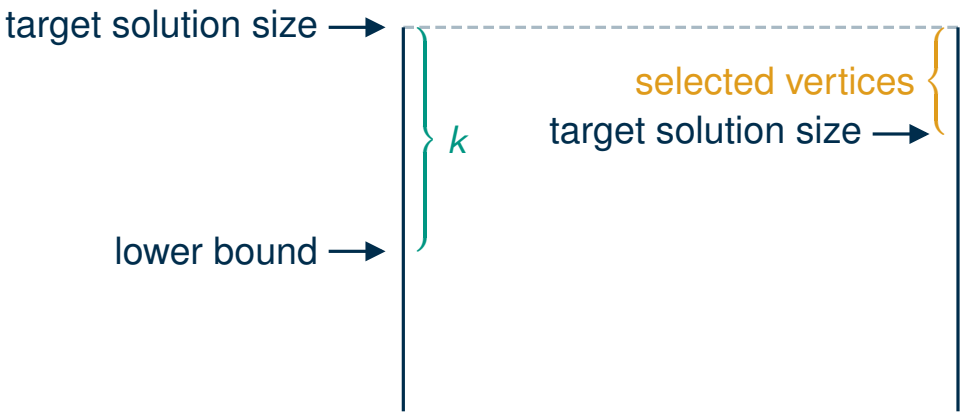
Reduce: if the LP relaxation has a solution with $G_{\frac{1}{2}} \subsetneq G$, reduce to $G_{\frac{1}{2}}$

Branch: for an edge uv, consider the child instances $G - u$ and $G - v$

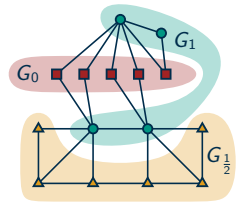
Problem: VERTEX COVER
Given a graph $G = (V, E)$ and a parameter k.
Does G have a vertex cover of size k?

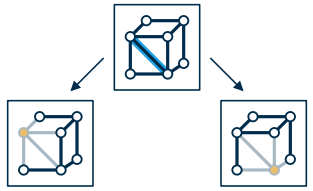


Problem: VERTEX COVER ABOVE LP
Given a graph $G = (V, E)$ and a parameter k.
Does G have a vertex cover of size $\ell_{\text{LP}}(G)+k$?



Branch-and-reduce: How does k change?

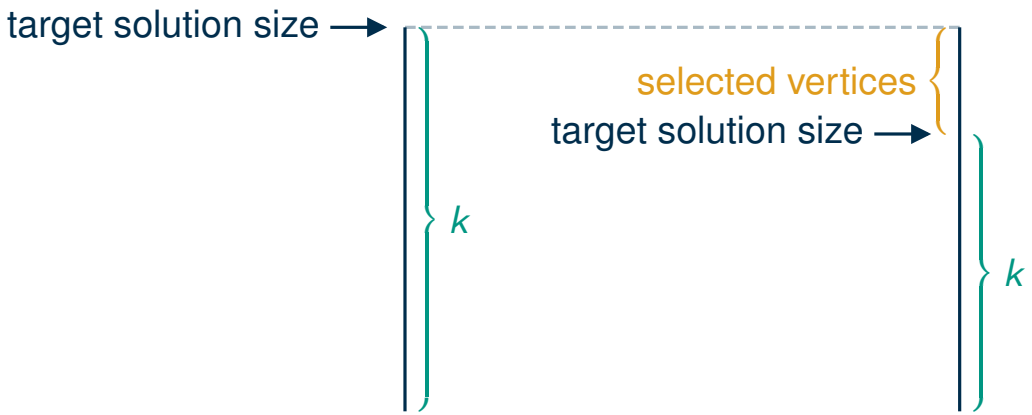




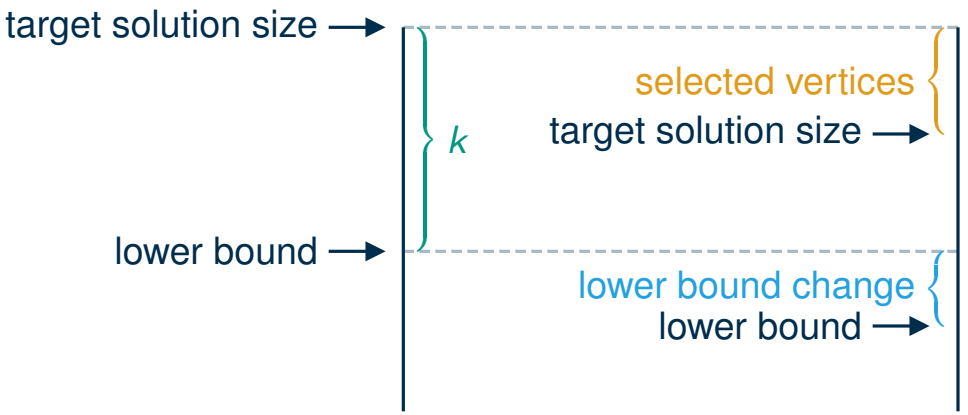
Reduce: if the LP relaxation has a solution with $G_{\frac{1}{2}} \subsetneq G$, reduce to $G_{\frac{1}{2}}$

Branch: for an edge uv, consider the child instances $G - u$ and $G - v$

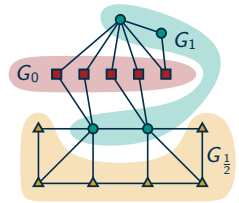
Problem: VERTEX COVER
Given a graph $G = (V, E)$ and a parameter k.
Does G have a vertex cover of size k?

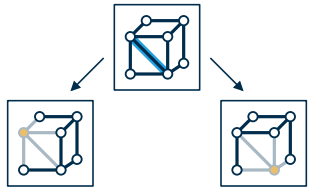


Problem: VERTEX COVER ABOVE LP
Given a graph $G = (V, E)$ and a parameter k.
Does G have a vertex cover of size $\ell_{\text{LP}}(G)+k$?



Branch-and-reduce: How does k change?

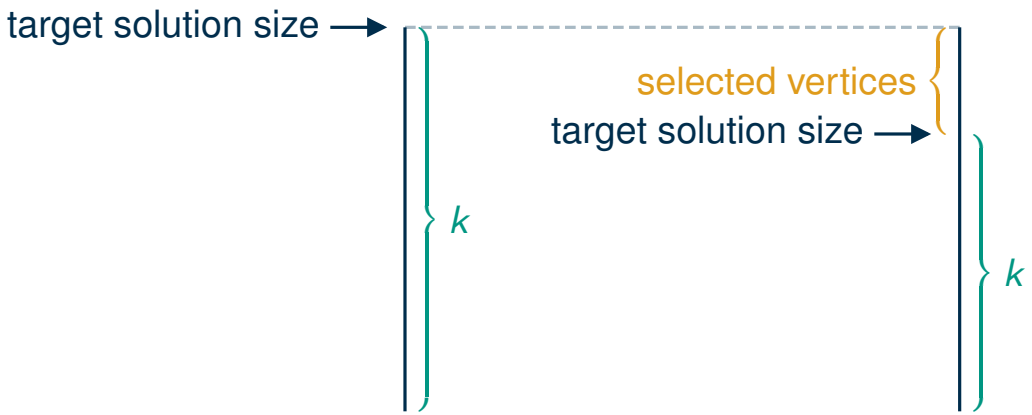




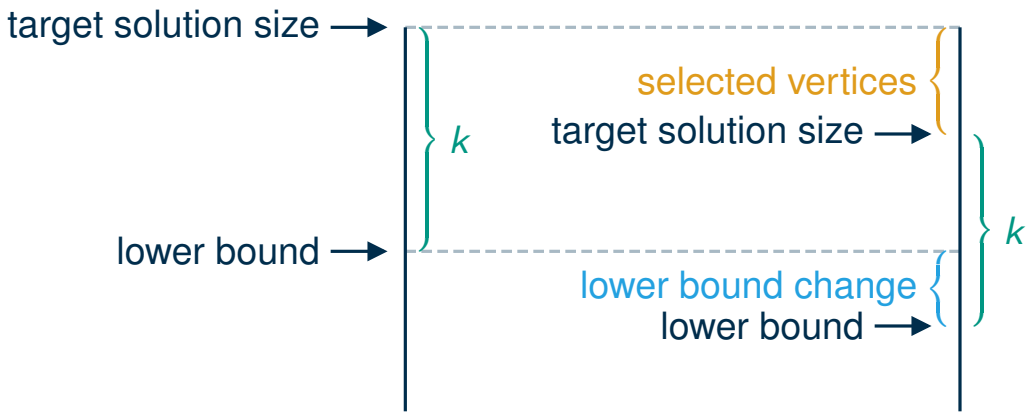
Reduce: if the LP relaxation has a solution with $G_{\frac{1}{2}} \subsetneq G$, reduce to $G_{\frac{1}{2}}$

Branch: for an edge uv, consider the child instances $G - u$ and $G - v$

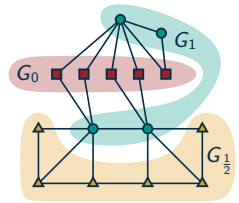
Problem: VERTEX COVER
Given a graph $G = (V, E)$ and a parameter k.
Does G have a vertex cover of size k?

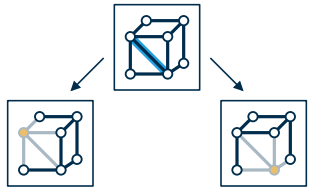


Problem: VERTEX COVER ABOVE LP
Given a graph $G = (V, E)$ and a parameter k.
Does G have a vertex cover of size $\ell_{\text{LP}}(G)+k$?



Branch-and-reduce: How does k change?

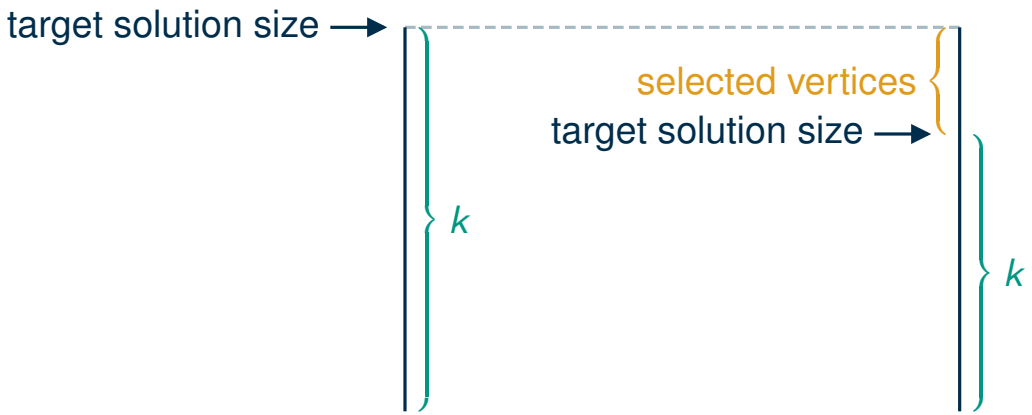




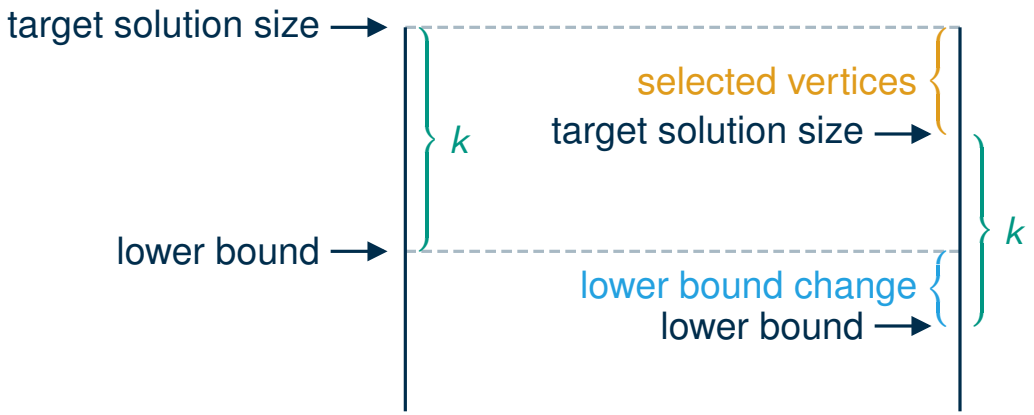
Reduce: if the LP relaxation has a solution with $G_{\frac{1}{2}} \subsetneq G$, reduce to $G_{\frac{1}{2}}$

Branch: for an edge uv, consider the child instances $G - u$ and $G - v$

Problem: VERTEX COVER
Given a graph $G = (V, E)$ and a parameter k.
Does G have a vertex cover of size k?



Problem: VERTEX COVER ABOVE LP
Given a graph $G = (V, E)$ and a parameter k.
Does G have a vertex cover of size $\ell_{\text{LP}}(G)+k$?

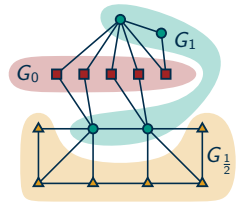


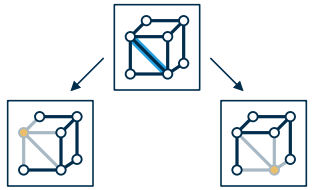
New parameter: depends on number of selected vertices and on how the lower bound changes

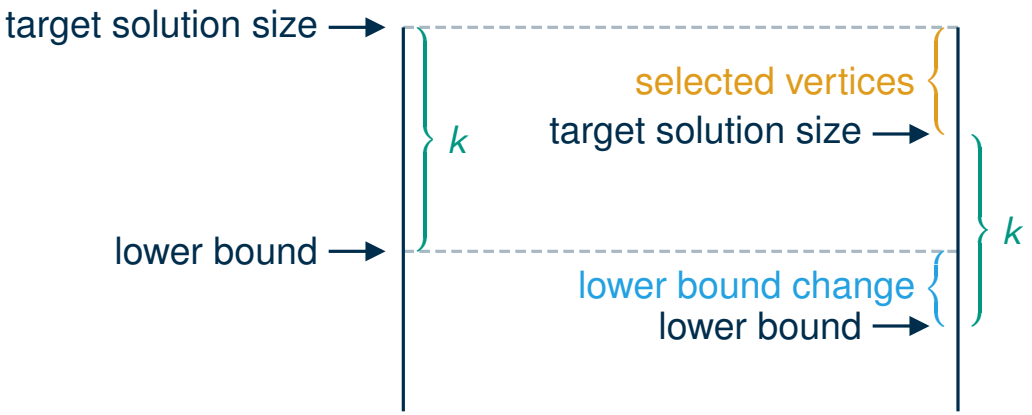
Branch-and-reduce: How does ℓ_{LP} change?

Reduce: if the LP relaxation has a solution with $G_{\frac{1}{2}} \subsetneq G$, reduce to $G_{\frac{1}{2}}$

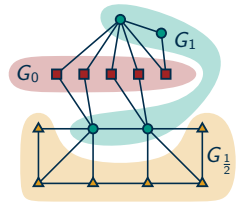
Branch: for an edge uv, consider the child instances $G - u$ and $G - v$

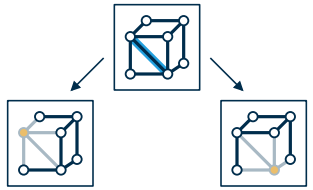






Branch-and-reduce: How does ℓ_{LP} change?



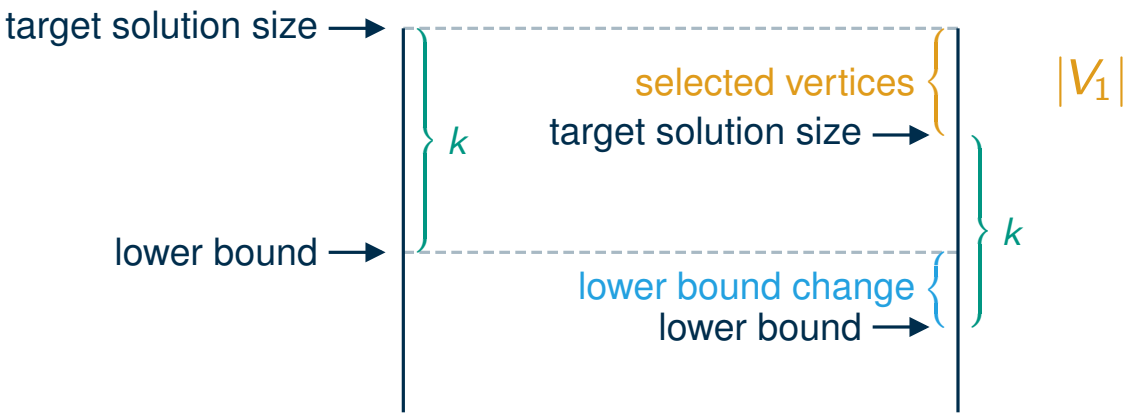


Reduce: if the LP relaxation has a solution with $G_{\frac{1}{2}} \subsetneq G$, reduce to $G_{\frac{1}{2}}$

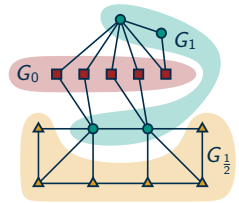
Branch: for an edge uv, consider the child instances $G - u$ and $G - v$

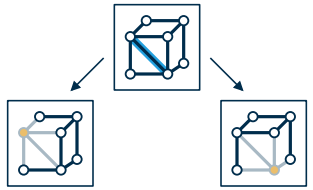
Reduction rule

- we add $|V_1|$ vertices to the vertex cover



Branch-and-reduce: How does ℓ_{LP} change?



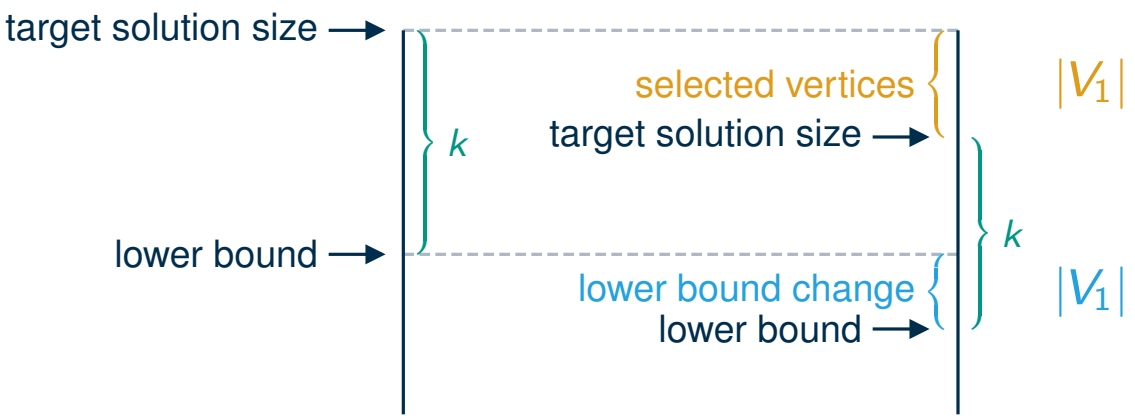


Reduce: if the LP relaxation has a solution with $G_{\frac{1}{2}} \subsetneq G$, reduce to $G_{\frac{1}{2}}$

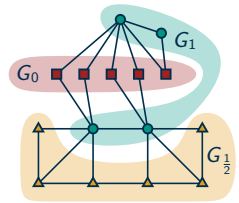
Branch: for an edge uv, consider the child instances $G - u$ and $G - v$

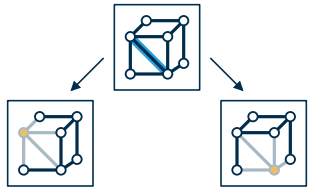
Reduction rule

- we add $|V_1|$ vertices to the vertex cover
- $\ell_{\text{LP}}(G_{\frac{1}{2}}) = \ell_{\text{LP}}(G) - |V_1|$



Branch-and-reduce: How does ℓ_{LP} change?



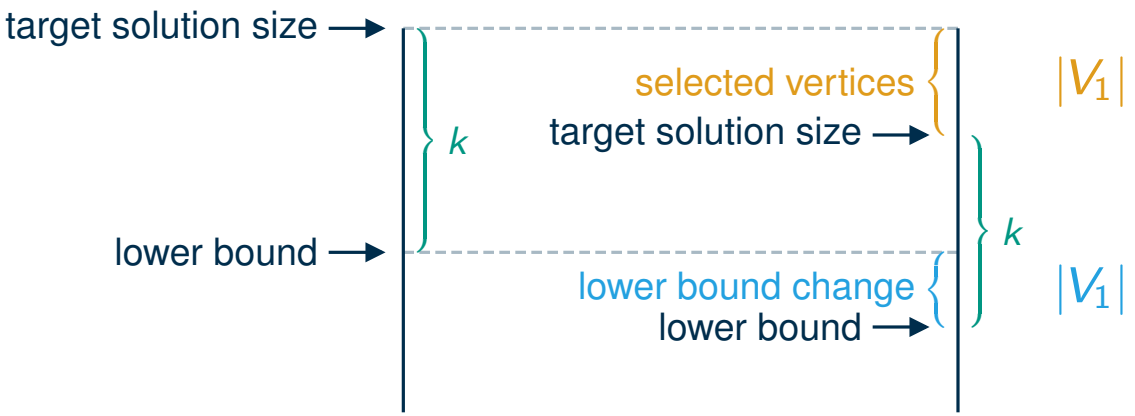


Reduce: if the LP relaxation has a solution with $G_{\frac{1}{2}} \subsetneq G$, reduce to $G_{\frac{1}{2}}$

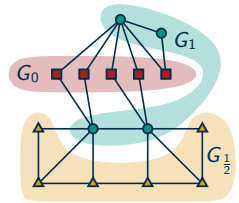
Branch: for an edge uv, consider the child instances $G - u$ and $G - v$

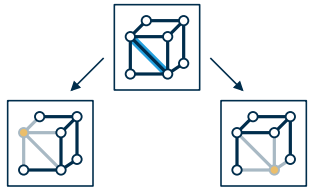
Reduction rule

- we add $|V_1|$ vertices to the vertex cover
- $\ell_{\text{LP}}(G_{\frac{1}{2}}) = \ell_{\text{LP}}(G) - |V_1|$
- the parameter k remains unchanged



Branch-and-reduce: How does ℓ_{LP} change?





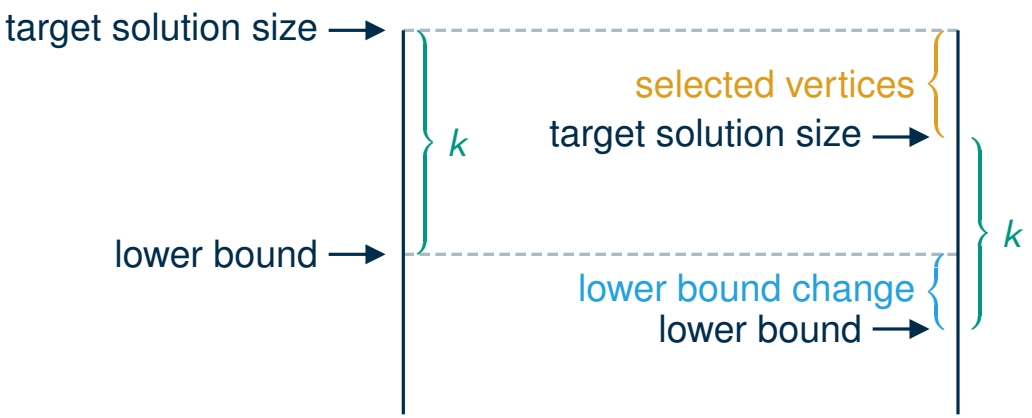
Reduce: if the LP relaxation has a solution with $G_{\frac{1}{2}} \subsetneq G$, reduce to $G_{\frac{1}{2}}$

Branch: for an edge uv, consider the child instances $G - u$ and $G - v$

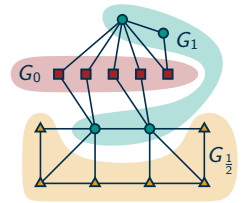
Reduction rule

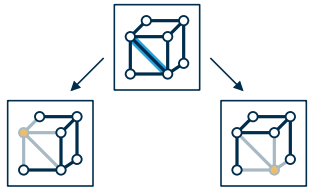
- we add $|V_1|$ vertices to the vertex cover
- $\ell_{\text{LP}}(G_{\frac{1}{2}}) = \ell_{\text{LP}}(G) - |V_1|$
- the parameter k remains unchanged

Branching rule



Branch-and-reduce: How does ℓ_{LP} change?





Reduce: if the LP relaxation has a solution with $G_{\frac{1}{2}} \subsetneq G$, reduce to $G_{\frac{1}{2}}$

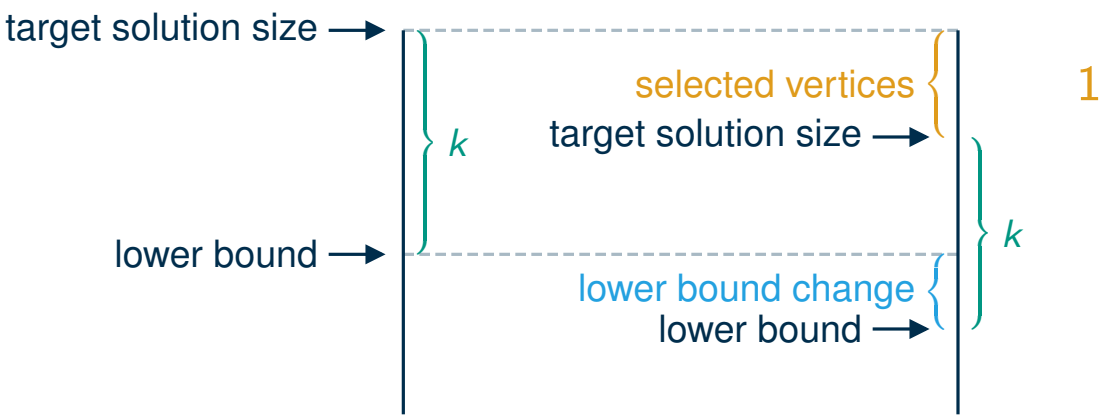
Branch: for an edge uv, consider the child instances $G - u$ and $G - v$

Reduction rule

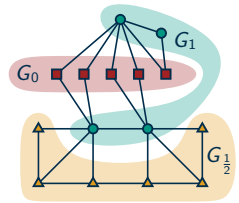
- we add $|V_1|$ vertices to the vertex cover
- $\ell_{\text{LP}}(G_{\frac{1}{2}}) = \ell_{\text{LP}}(G) - |V_1|$
- the parameter k remains unchanged

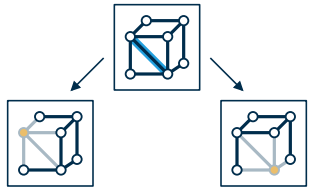
Branching rule

- 1 vertex is added to the vertex cover



Branch-and-reduce: How does ℓ_{LP} change?





Reduce: if the LP relaxation has a solution with $G_{\frac{1}{2}} \subsetneq G$, reduce to $G_{\frac{1}{2}}$

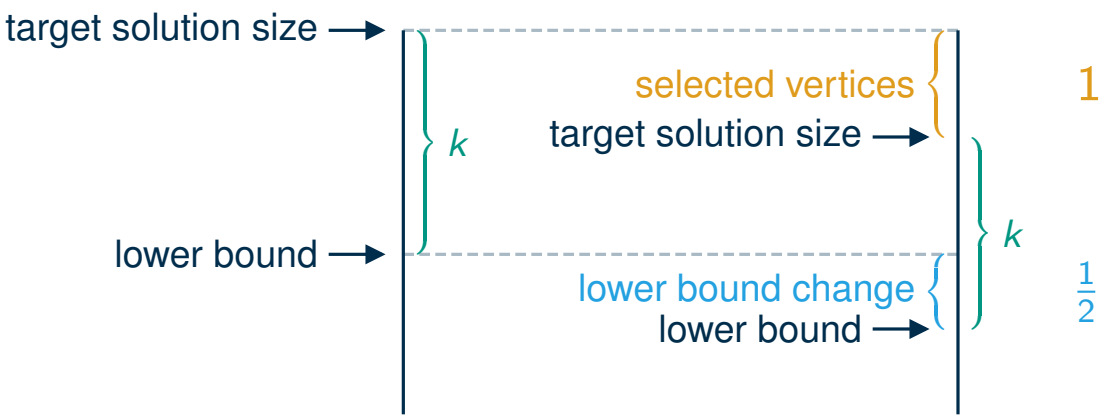
Branch: for an edge uv, consider the child instances $G - u$ and $G - v$

Reduction rule

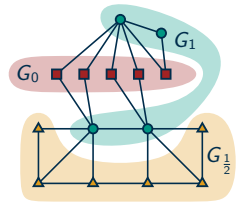
- we add $|V_1|$ vertices to the vertex cover
- $\ell_{\text{LP}}(G_{\frac{1}{2}}) = \ell_{\text{LP}}(G) - |V_1|$
- the parameter k remains unchanged

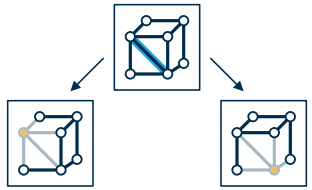
Branching rule

- 1 vertex is added to the vertex cover
- next slide: ℓ_{LP} shrinks by $\frac{1}{2}$



Branch-and-reduce: How does ℓ_{LP} change?





Reduce: if the LP relaxation has a solution with $G_{\frac{1}{2}} \subsetneq G$, reduce to $G_{\frac{1}{2}}$

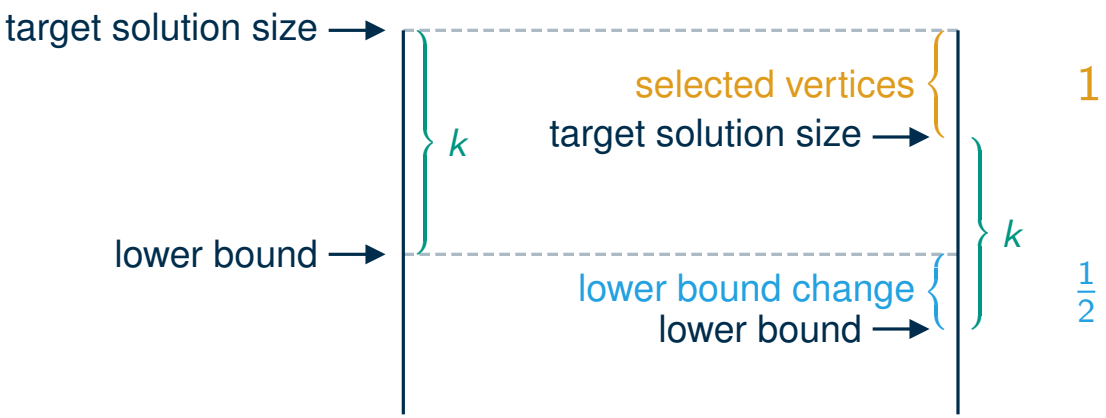
Branch: for an edge uv, consider the child instances $G - u$ and $G - v$

Reduction rule

- we add $|V_1|$ vertices to the vertex cover
- $\ell_{\text{LP}}(G_{\frac{1}{2}}) = \ell_{\text{LP}}(G) - |V_1|$
- the parameter k remains unchanged

Branching rule

- 1 vertex is added to the vertex cover
- next slide: ℓ_{LP} shrinks by $\frac{1}{2}$
- this implies: the parameter k shrinks by $\frac{1}{2}$



How does $\ell_{\text{LP}}(G - v)$ compare to $\ell_{\text{LP}}(G)$?

Lemma
We cannot apply the reduction rule (i.e., setting $x_v = \frac{1}{2}$ for all $v \in V$ is the only LP-solution) if and only if $\ell_{\text{LP}}(G - v) = \ell_{\text{LP}}(G) - \frac{1}{2}$ for all $v \in V$.

How does $\ell_{\text{LP}}(G - v)$ compare to $\ell_{\text{LP}}(G)$?

Lemma
We cannot apply the reduction rule (i.e., setting $x_v = \frac{1}{2}$ for all $v \in V$ is the only LP-solution) if and only if $\ell_{\text{LP}}(G - v) = \ell_{\text{LP}}(G) - \frac{1}{2}$ for all $v \in V$.

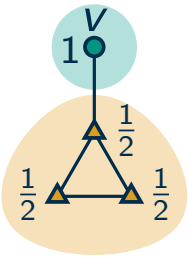
Proof for "$\Leftarrow$": $x_v = \frac{1}{2}$ is only LP-solution $\Leftarrow \ell_{\text{LP}}(G - v) = \ell_{\text{LP}}(G) - \frac{1}{2}$ (for all $v \in V$)

How does $\ell_{\text{LP}}(G - v)$ compare to $\ell_{\text{LP}}(G)$?

Lemma
We cannot apply the reduction rule (i.e., setting $x_v = \frac{1}{2}$ for all $v \in V$ is the only LP-solution) if and only if $\ell_{\text{LP}}(G - v) = \ell_{\text{LP}}(G) - \frac{1}{2}$ for all $v \in V$.

Proof for "$\Leftarrow$": $x_v = \frac{1}{2}$ is only LP-solution $\Leftarrow \ell_{\text{LP}}(G - v) = \ell_{\text{LP}}(G) - \frac{1}{2}$ (for all $v \in V$)

- assume there is a LP-solution with $x_v = 1$ for a vertex v

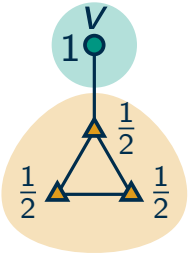


How does $\ell_{\text{LP}}(G - v)$ compare to $\ell_{\text{LP}}(G)$?

Lemma
We cannot apply the reduction rule (i.e., setting $x_v = \frac{1}{2}$ for all $v \in V$ is the only LP-solution) if and only if $\ell_{\text{LP}}(G - v) = \ell_{\text{LP}}(G) - \frac{1}{2}$ for all $v \in V$.

Proof for "$\Leftarrow$": $x_v = \frac{1}{2}$ is only LP-solution $\Leftarrow \ell_{\text{LP}}(G - v) = \ell_{\text{LP}}(G) - \frac{1}{2}$ (for all $v \in V$)

- assume there is a LP-solution with $x_v = 1$ for a vertex v
- the same values (without x_v) yield LP-solution for $G - v$

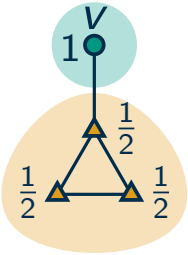


How does $\ell_{\text{LP}}(G - v)$ compare to $\ell_{\text{LP}}(G)$?

Lemma
We cannot apply the reduction rule (i.e., setting $x_v = \frac{1}{2}$ for all $v \in V$ is the only LP-solution) if and only if $\ell_{\text{LP}}(G - v) = \ell_{\text{LP}}(G) - \frac{1}{2}$ for all $v \in V$.

Proof for "$\Leftarrow$": $x_v = \frac{1}{2}$ is only LP-solution $\Leftarrow \ell_{\text{LP}}(G - v) = \ell_{\text{LP}}(G) - \frac{1}{2}$ (for all $v \in V$)

- assume there is a LP-solution with $x_v = 1$ for a vertex v
- the same values (without x_v) yield LP-solution for $G - v$
- so: $G - v$ has an LP-solution of size $\ell_{\text{LP}}(G) - 1$

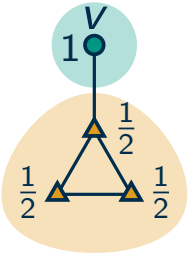


How does $\ell_{\text{LP}}(G - v)$ compare to $\ell_{\text{LP}}(G)$?

Lemma
We cannot apply the reduction rule (i.e., setting $x_v = \frac{1}{2}$ for all $v \in V$ is the only LP-solution) if and only if $\ell_{\text{LP}}(G - v) = \ell_{\text{LP}}(G) - \frac{1}{2}$ for all $v \in V$.

Proof for "$\Leftarrow$": $x_v = \frac{1}{2}$ is only LP-solution $\Leftarrow \ell_{\text{LP}}(G - v) = \ell_{\text{LP}}(G) - \frac{1}{2}$ (for all $v \in V$)

- assume there is a LP-solution with $x_v = 1$ for a vertex v
- the same values (without x_v) yield LP-solution for $G - v$
- so: $G - v$ has an LP-solution of size $\ell_{\text{LP}}(G) - 1$



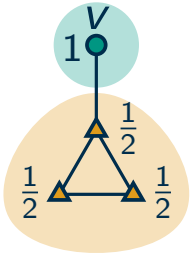
Proof for "$\Rightarrow$": $x_v = \frac{1}{2}$ is only LP-solution $\Rightarrow \ell_{\text{LP}}(G - v) = \ell_{\text{LP}}(G) - \frac{1}{2}$ (for all $v \in V$)

How does $\ell_{\text{LP}}(G - v)$ compare to $\ell_{\text{LP}}(G)$?

Lemma
We cannot apply the reduction rule (i.e., setting $x_v = \frac{1}{2}$ for all $v \in V$ is the only LP-solution) if and only if $\ell_{\text{LP}}(G - v) = \ell_{\text{LP}}(G) - \frac{1}{2}$ for all $v \in V$.

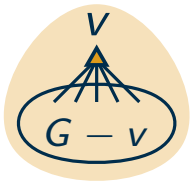
Proof for "$\Leftarrow$": $x_v = \frac{1}{2}$ is only LP-solution $\Leftarrow \ell_{\text{LP}}(G - v) = \ell_{\text{LP}}(G) - \frac{1}{2}$ (for all $v \in V$)

- assume there is a LP-solution with $x_v = 1$ for a vertex v
- the same values (without x_v) yield LP-solution for $G - v$
- so: $G - v$ has an LP-solution of size $\ell_{\text{LP}}(G) - 1$



Proof for "$\Rightarrow$": $x_v = \frac{1}{2}$ is only LP-solution $\Rightarrow \ell_{\text{LP}}(G - v) = \ell_{\text{LP}}(G) - \frac{1}{2}$ (for all $v \in V$)

- obvious: $\ell_{\text{LP}}(G - v) \leq \ell_{\text{LP}}(G) - \frac{1}{2}$ (all $\frac{1}{2}$ with one less vertex)

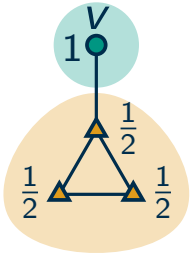


How does $\ell_{\text{LP}}(G - v)$ compare to $\ell_{\text{LP}}(G)$?

Lemma
We cannot apply the reduction rule (i.e., setting $x_v = \frac{1}{2}$ for all $v \in V$ is the only LP-solution) if and only if $\ell_{\text{LP}}(G - v) = \ell_{\text{LP}}(G) - \frac{1}{2}$ for all $v \in V$.

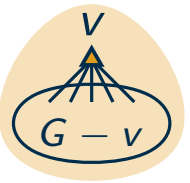
Proof for "$\Leftarrow$": $x_v = \frac{1}{2}$ is only LP-solution $\Leftarrow \ell_{\text{LP}}(G - v) = \ell_{\text{LP}}(G) - \frac{1}{2}$ (for all $v \in V$)

- assume there is a LP-solution with $x_v = 1$ for a vertex v
- the same values (without x_v) yield LP-solution for $G - v$
- so: $G - v$ has an LP-solution of size $\ell_{\text{LP}}(G) - 1$



Proof for "$\Rightarrow$": $x_v = \frac{1}{2}$ is only LP-solution $\Rightarrow \ell_{\text{LP}}(G - v) = \ell_{\text{LP}}(G) - \frac{1}{2}$ (for all $v \in V$)

- obvious: $\ell_{\text{LP}}(G - v) \leq \ell_{\text{LP}}(G) - \frac{1}{2}$ (all $\frac{1}{2}$ with one less vertex)
- if $\ell_{\text{LP}}(G - v) < \ell_{\text{LP}}(G) - \frac{1}{2}$: consider LP-solution for $G - v$ with value $\leq \ell_{\text{LP}}(G) - 1$

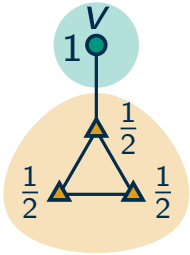


How does $\ell_{\text{LP}}(G - v)$ compare to $\ell_{\text{LP}}(G)$?

Lemma
We cannot apply the reduction rule (i.e., setting $x_v = \frac{1}{2}$ for all $v \in V$ is the only LP-solution) if and only if $\ell_{\text{LP}}(G - v) = \ell_{\text{LP}}(G) - \frac{1}{2}$ for all $v \in V$.

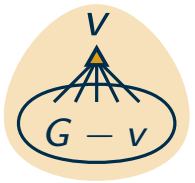
Proof for "⇐": $x_v = \frac{1}{2}$ is only LP-solution $\Leftarrow \ell_{\text{LP}}(G - v) = \ell_{\text{LP}}(G) - \frac{1}{2}$ (for all $v \in V$)

- assume there is a LP-solution with $x_v = 1$ for a vertex v
- the same values (without x_v) yield LP-solution for $G - v$
- so: $G - v$ has an LP-solution of size $\ell_{\text{LP}}(G) - 1$



Proof for "⇒": $x_v = \frac{1}{2}$ is only LP-solution $\Rightarrow \ell_{\text{LP}}(G - v) = \ell_{\text{LP}}(G) - \frac{1}{2}$ (for all $v \in V$)

- obvious: $\ell_{\text{LP}}(G - v) \leq \ell_{\text{LP}}(G) - \frac{1}{2}$ (all $\frac{1}{2}$ with one less vertex)
- if $\ell_{\text{LP}}(G - v) < \ell_{\text{LP}}(G) - \frac{1}{2}$: consider LP-solution for $G - v$ with value $\leq \ell_{\text{LP}}(G) - 1$
- add v with $x_v = 1$ to it $\rightarrow$ optimal LP-solution for G



Running time

Lemma
We cannot apply the reduction rule (i.e., setting $x_v = \frac{1}{2}$ for all $v \in V$ is the only LP-solution) if and only if $\ell_{\text{LP}}(G - v) = \ell_{\text{LP}}(G) - \frac{1}{2}$ for all $v \in V$.

Branching rule

- 1 vertex is added to the vertex cover
- previous slide: ℓ_{LP} shrinks by $\frac{1}{2}$
- this implies: the parameter k shrinks by $\frac{1}{2}$

Running time

Lemma
We cannot apply the reduction rule (i.e., setting $x_v = \frac{1}{2}$ for all $v \in V$ is the only LP-solution) if and only if $\ell_{\text{LP}}(G - v) = \ell_{\text{LP}}(G) - \frac{1}{2}$ for all $v \in V$.

Branching rule

- 1 vertex is added to the vertex cover
- previous slide: ℓ_{LP} shrinks by $\frac{1}{2}$
- this implies: the parameter k shrinks by $\frac{1}{2}$

} search tree height $\leq 2k \Rightarrow 2^{2k} = 4^k$ leaves

Running time

Lemma
We cannot apply the reduction rule (i.e., setting $x_v = \frac{1}{2}$ for all $v \in V$ is the only LP-solution) if and only if $\ell_{\text{LP}}(G - v) = \ell_{\text{LP}}(G) - \frac{1}{2}$ for all $v \in V$.

Branching rule

- 1 vertex is added to the vertex cover
- previous slide: ℓ_{LP} shrinks by $\frac{1}{2}$
- this implies: the parameter k shrinks by $\frac{1}{2}$

} search tree height $\leq 2k \Rightarrow 2^{2k} = 4^k$ leaves

Reduction rule

How can we check that **every** LP-solution has $x_v = \frac{1}{2}$ for all $v \in V$?

Running time

Lemma
We cannot apply the reduction rule (i.e., setting $x_v = \frac{1}{2}$ for all $v \in V$ is the only LP-solution) if and only if $\ell_{\text{LP}}(G - v) = \ell_{\text{LP}}(G) - \frac{1}{2}$ for all $v \in V$.

Branching rule

- 1 vertex is added to the vertex cover
- previous slide: ℓ_{LP} shrinks by $\frac{1}{2}$
- this implies: the parameter k shrinks by $\frac{1}{2}$

} search tree height $\leq 2k \Rightarrow 2^{2k} = 4^k$ leaves

Reduction rule

- solve the LP for $G - v$ for every $v \in V$

Running time

Lemma
We cannot apply the reduction rule (i.e., setting $x_v = \frac{1}{2}$ for all $v \in V$ is the only LP-solution) if and only if $\ell_{\text{LP}}(G - v) = \ell_{\text{LP}}(G) - \frac{1}{2}$ for all $v \in V$.

Branching rule

- 1 vertex is added to the vertex cover
- previous slide: ℓ_{LP} shrinks by $\frac{1}{2}$
- this implies: the parameter k shrinks by $\frac{1}{2}$

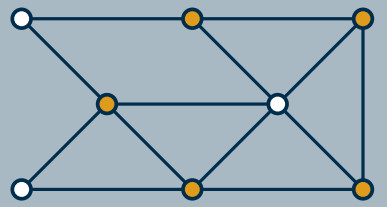
} search tree height $\leq 2k \Rightarrow 2^{2k} = 4^k$ leaves

Reduction rule

- solve the LP for $G - v$ for every $v \in V$
- lemma: reduction applicable $\Rightarrow$ we find $v \in V$ with $\ell_{\text{LP}}(G - v) < \ell_{\text{LP}}(G) - \frac{1}{2}$
- note: this means that there is an optimal LP-solution with $x_v = 1$

Wrap-Up: Vertex Cover above LP

Problem: Vertex Cover above LP
Given a graph $G = (V, E)$ and a parameter k. Does G have a vertex cover of size $\ell_{\text{LP}}(G) + k$?



(vertex set $V' \subseteq V$ with $e \cap V' \neq \emptyset$ for all $e \in E$)

Theorem
There is an FPT algorithm for Vertex Cover above LP with running time $4^k \cdot n^{O(1)}$.

Branch-and-reduce

- reduce if possible using LP-relaxation
- if reduction not possible: branching makes progress (even above lower bound)

How long is the break?

$$\begin{aligned} \text{min.:} \quad & B + r + e + a + k \\ \text{such that:} \quad & B + k \geq 2 \\ & r - e \geq 2 \\ & 2e + a \geq 1 \end{aligned}$$

How long is the break?

$$\begin{aligned} \text{min.:} \quad & B + r + e + a + k \\ \text{such that:} \quad & B + k \geq 2 \\ & r - e \geq 2 \\ & 2e + a \geq 1 \end{aligned}$$

Optimal solution (value 5):

$B = 1, r = 2.5, e = 0.5, a = 0, k = 1$

How long is the break?

$$\begin{aligned} \text{min.:} \quad & B + r + e + a + k \\ \text{such that:} \quad & B + k \geq 2 \\ & r - e \geq 2 \\ & 2e + a \geq 1 \end{aligned}$$

Optimal solution (value 5):
$B = 1, r = 2.5, e = 0.5, a = 0, k = 1$

Why is there no better solution?

Finding upper bounds

Goal: Find an upper bound for the optimal solution

maximize: $2x_1 + 3x_2$

such that: $4x_1 + 8x_2 \leq 12$

$2x_1 + 1x_2 \leq 3$

$3x_1 + 2x_2 \leq 4$

$x_1, x_2 \geq 0$

Finding upper bounds

Goal: Find an upper bound for the optimal solution

- simple bound: 12

$2x_1 + 3x_2 \leq 4x_1 + 8x_2 \leq 12$

maximize: $2x_1 + 3x_2$

such that: $4x_1 + 8x_2 \leq 12$

$2x_1 + 1x_2 \leq 3$

$3x_1 + 2x_2 \leq 4$

$x_1, x_2 \geq 0$

Finding upper bounds

Goal: Find an upper bound for the optimal solution

- simple bound: 12
- better: 6

$$2x_1 + 3x_2 \leq 4x_1 + 8x_2 \leq 12$$

$$2x_1 + 3x_2 \leq \frac{4x_1+8x_2}{2} \leq \frac{12}{2}$$

$\frac{1}{2}$

maximize: $2x_1 + 3x_2$

such that: $4x_1 + 8x_2 \leq 12$

$2x_1 + 1x_2 \leq 3$

$3x_1 + 2x_2 \leq 4$

$x_1, x_2 \geq 0$

Finding upper bounds

Goal: Find an upper bound for the optimal solution

- simple bound: 12
- better: 6
- combine constraints: 5

$$2x_1 + 3x_2 \leq 4x_1 + 8x_2 \leq 12$$

$$2x_1 + 3x_2 \leq \frac{4x_1+8x_2}{2} \leq \frac{12}{2}$$

$$2x_1 + 3x_2 \leq \frac{4x_1+8x_2}{3} + \frac{2x_1+1x_2}{3} \leq \frac{12}{3} + \frac{3}{3}$$

maximize: $2x_1 + 3x_2$

such that: $4x_1 + 8x_2 \leq 12$

$2x_1 + 1x_2 \leq 3$

$3x_1 + 2x_2 \leq 4$

$x_1, x_2 \geq 0$

Finding upper bounds

Goal: Find an upper bound for the optimal solution

- simple bound: 12
- better: 6
- combine constraints: 5

$$2x_1 + 3x_2 \leq 4x_1 + 8x_2 \leq 12$$

$$2x_1 + 3x_2 \leq \frac{4x_1+8x_2}{2} \leq \frac{12}{2}$$

$$2x_1 + 3x_2 \leq \frac{4x_1+8x_2}{3} + \frac{2x_1+1x_2}{3} \leq \frac{12}{3} + \frac{3}{3}$$

maximize: $2x_1 + 3x_2$

such that:

$4x_1 + 8x_2 \leq 12$ ($\frac{1}{2}$)

$2x_1 + 1x_2 \leq 3$ ($\frac{1}{3}$)

$3x_1 + 2x_2 \leq 4$ ($\frac{1}{3}$)

$x_1, x_2 \geq 0$

Systematically combine constraints with factors y_1, y_2, and y_3

- resulting inequality:

$$y_1(4x_1 + 8x_2) + y_2(2x_1 + 1x_2) + y_3(3x_1 + 2x_2) \leq 12y_1 + 3y_2 + 4y_3$$

Finding upper bounds

Goal: Find an upper bound for the optimal solution

- simple bound: 12
- better: 6
- combine constraints: 5

$$2x_1 + 3x_2 \leq 4x_1 + 8x_2 \leq 12$$

$$2x_1 + 3x_2 \leq \frac{4x_1+8x_2}{2} \leq \frac{12}{2}$$

$$2x_1 + 3x_2 \leq \frac{4x_1+8x_2}{3} + \frac{2x_1+1x_2}{3} \leq \frac{12}{3} + \frac{3}{3}$$

maximize: $2x_1 + 3x_2$

such that:

$4x_1 + 8x_2 \leq 12$ ($\cdot \frac{1}{2}$)

$2x_1 + 1x_2 \leq 3$ ($\cdot \frac{1}{3}$)

$3x_1 + 2x_2 \leq 4$ ($\cdot \frac{1}{3}$)

$x_1, x_2 \geq 0$

Systematically combine constraints with factors y_1, y_2, and y_3

- resulting inequality:
- rearranging yields:

$$y_1(4x_1 + 8x_2) + y_2(2x_1 + 1x_2) + y_3(3x_1 + 2x_2) \leq 12y_1 + 3y_2 + 4y_3$$

$$(4y_1 + 2y_2 + 3y_3)x_1 + (8y_1 + 1y_2 + 2y_3)x_2 \leq 12y_1 + 3y_2 + 4y_3$$

Finding upper bounds

Goal: Find an upper bound for the optimal solution

- simple bound: 12
- better: 6
- combine constraints: 5

$$2x_1 + 3x_2 \leq 4x_1 + 8x_2 \leq 12$$

$$2x_1 + 3x_2 \leq \frac{4x_1+8x_2}{2} \leq \frac{12}{2}$$

$$2x_1 + 3x_2 \leq \frac{4x_1+8x_2}{3} + \frac{2x_1+1x_2}{3} \leq \frac{12}{3} + \frac{3}{3}$$

maximize: $2x_1 + 3x_2$

such that:

$4x_1 + 8x_2 \leq 12$ (factor $\frac{1}{2}$)

$2x_1 + 1x_2 \leq 3$ (factor $\frac{1}{3}$)

$3x_1 + 2x_2 \leq 4$ (factor $\frac{1}{3}$)

$x_1, x_2 \geq 0$

Systematically combine constraints with factors y_1, y_2, and y_3

- resulting inequality: $y_1(4x_1 + 8x_2) + y_2(2x_1 + 1x_2) + y_3(3x_1 + 2x_2) \leq 12y_1 + 3y_2 + 4y_3$
- rearranging yields: $2x_1 + 3x_2 \leq \underbrace{(4y_1 + 2y_2 + 3y_3)}_{\geq 2}x_1 + \underbrace{(8y_1 + 1y_2 + 2y_3)}_{\geq 3}x_2 \leq 12y_1 + 3y_2 + 4y_3$

(required for the first inequality)

Finding upper bounds

Goal: Find an upper bound for the optimal solution

- simple bound: 12
- better: 6
- combine constraints: 5

$$2x_1 + 3x_2 \leq 4x_1 + 8x_2 \leq 12$$

$$2x_1 + 3x_2 \leq \frac{4x_1+8x_2}{2} \leq \frac{12}{2} \quad \left(\tfrac{1}{2}\right)$$

$$2x_1 + 3x_2 \leq \frac{4x_1+8x_2}{3} + \frac{2x_1+1x_2}{3} \leq \frac{12}{3} + \frac{3}{3} \quad \left(\tfrac{1}{3}, \tfrac{1}{3}\right)$$

maximize: $2x_1 + 3x_2$

such that:

$$4x_1 + 8x_2 \leq 12$$
$$2x_1 + 1x_2 \leq 3$$
$$3x_1 + 2x_2 \leq 4$$
$$x_1, x_2 \geq 0$$

Systematically combine constraints with factors y_1, y_2, and y_3

- resulting inequality:

$$y_1(4x_1 + 8x_2) + y_2(2x_1 + 1x_2) + y_3(3x_1 + 2x_2) \leq 12y_1 + 3y_2 + 4y_3$$

- rearranging yields:

$$2x_1 + 3x_2 \leq \underbrace{(4y_1 + 2y_2 + 3y_3)}_{\geq 2}x_1 + \underbrace{(8y_1 + 1y_2 + 2y_3)}_{\geq 3}x_2 \leq \underbrace{12y_1 + 3y_2 + 4y_3}_{\text{minimize}}$$

(required for the first inequality)

(for the best upper bound)

Finding upper bounds

Goal: Find an upper bound for the optimal solution

- simple bound: 12
- better: 6
- combine constraints: 5

$$2x_1 + 3x_2 \leq 4x_1 + 8x_2 \leq 12$$

$$2x_1 + 3x_2 \leq \frac{4x_1+8x_2}{2} \leq \frac{12}{2}$$

$$2x_1 + 3x_2 \leq \frac{4x_1+8x_2}{3} + \frac{2x_1+1x_2}{3} \leq \frac{12}{3} + \frac{3}{3}$$

maximize: $2x_1 + 3x_2$

such that: $4x_1 + 8x_2 \leq 12$

$2x_1 + 1x_2 \leq 3$

$3x_1 + 2x_2 \leq 4$

$x_1, x_2 \geq 0$

Systematically combine constraints with factors y_1, y_2, and y_3

- resulting inequality: $y_1(4x_1 + 8x_2) + y_2(2x_1 + 1x_2) + y_3(3x_1 + 2x_2) \leq 12y_1 + 3y_2 + 4y_3$
- rearranging yields: $2x_1 + 3x_2 \leq \underbrace{(4y_1 + 2y_2 + 3y_3)}_{\geq 2}x_1 + \underbrace{(8y_1 + 1y_2 + 2y_3)}_{\geq 3}x_2 \leq \underbrace{12y_1 + 3y_2 + 4y_3}_{\text{minimize}}$

(required for the first inequality)

(for the best upper bound)

Dual program

minimize: $12y_1 + 3y_2 + 4y_3$

such that: $4y_1 + 2y_2 + 3y_3 \geq 2$

$8y_1 + 1y_2 + 2y_3 \geq 3$

Finding upper bounds

Goal: Find an upper bound for the optimal solution

- simple bound: 12
- better: 6
- combine constraints: 5

$$2x_1 + 3x_2 \leq 4x_1 + 8x_2 \leq 12$$

$$2x_1 + 3x_2 \leq \frac{4x_1+8x_2}{2} \leq \frac{12}{2} \quad \left(\tfrac{1}{2}\right)$$

$$2x_1 + 3x_2 \leq \frac{4x_1+8x_2}{3} + \frac{2x_1+1x_2}{3} \leq \frac{12}{3} + \frac{3}{3} \quad \left(\tfrac{1}{3}, \tfrac{1}{3}\right)$$

maximize: $2x_1 + 3x_2$

such that:

$$4x_1 + 8x_2 \leq 12$$
$$2x_1 + 1x_2 \leq 3$$
$$3x_1 + 2x_2 \leq 4$$
$$x_1, x_2 \geq 0$$

Systematically combine constraints with factors y_1, y_2, and y_3

- resulting inequality:

$$y_1(4x_1 + 8x_2) + y_2(2x_1 + 1x_2) + y_3(3x_1 + 2x_2) \leq 12y_1 + 3y_2 + 4y_3$$

- rearranging yields:

$$2x_1 + 3x_2 \leq \underbrace{(4y_1 + 2y_2 + 3y_3)}_{\geq 2}x_1 + \underbrace{(8y_1 + 1y_2 + 2y_3)}_{\geq 3}x_2 \leq \underbrace{12y_1 + 3y_2 + 4y_3}_{\text{minimize}}$$

(required for the first inequality)

(for the best upper bound)

Dual program

- additionally require $y1, y2, y3 \geq 0$

Why do we need this?

minimize: $12y_1 + 3y_2 + 4y_3$

such that:

$$4y_1 + 2y_2 + 3y_3 \geq 2$$
$$8y_1 + 1y_2 + 2y_3 \geq 3$$

Finding upper bounds

Goal: Find an upper bound for the optimal solution

- simple bound: 12
- better: 6
- combine constraints: 5

$$2x_1 + 3x_2 \leq 4x_1 + 8x_2 \leq 12$$

$$2x_1 + 3x_2 \leq \frac{4x_1+8x_2}{2} \leq \frac{12}{2}$$

$$2x_1 + 3x_2 \leq \frac{4x_1+8x_2}{3} + \frac{2x_1+1x_2}{3} \leq \frac{12}{3} + \frac{3}{3}$$

maximize: $2x_1 + 3x_2$

such that: $4x_1 + 8x_2 \leq 12$

$2x_1 + 1x_2 \leq 3$

$3x_1 + 2x_2 \leq 4$

$x_1, x_2 \geq 0$

Systematically combine constraints with factors y_1, y_2, and y_3

- resulting inequality: $y_1(4x_1 + 8x_2) + y_2(2x_1 + 1x_2) + y_3(3x_1 + 2x_2) \leq 12y_1 + 3y_2 + 4y_3$
- rearranging yields: $2x_1 + 3x_2 \leq \underbrace{(4y_1 + 2y_2 + 3y_3)}_{\geq 2}x_1 + \underbrace{(8y_1 + 1y_2 + 2y_3)}_{\geq 3}x_2 \leq \underbrace{12y_1 + 3y_2 + 4y_3}_{\text{minimize}}$

(required for the first inequality)

(for the best upper bound)

Dual program

- additionally require $y1, y2, y3 \geq 0$
- finds the smallest upper bound that can be achieved this way

Why do we need this?

minimize: $12y_1 + 3y_2 + 4y_3$

such that: $4y_1 + 2y_2 + 3y_3 \geq 2$

$8y_1 + 1y_2 + 2y_3 \geq 3$

Matrix form

Primal linear program

$$\begin{aligned} \text{maximize: } & 2x_1 + 3x_2 \\ \text{such that: } & 4x_1 + 8x_2 \leq 12 \\ & 2x_1 + 1x_2 \leq 3 \\ & 3x_1 + 2x_2 \leq 4 \\ & x_1, x_2 \geq 0 \end{aligned}$$

Dual linear program

$$\begin{aligned} \text{minimize: } & 12y_1 + 3y_2 + 4y_3 \\ \text{such that: } & 4y_1 + 2y_2 + 3y_3 \geq 2 \\ & 8y_1 + 1y_2 + 2y_3 \geq 3 \\ & y_1, y_2, y_3 \geq 0 \end{aligned}$$

Matrix form

Primal linear program

$$\begin{aligned} \text{maximize: } & 2x_1 + 3x_2 \\ \text{such that: } & 4x_1 + 8x_2 \leq 12 \\ & 2x_1 + 1x_2 \leq 3 \\ & 3x_1 + 2x_2 \leq 4 \\ & x_1, x_2 \geq 0 \end{aligned}$$

$$\max \begin{pmatrix} 2 & 3 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$

$$\text{with } \begin{pmatrix} 4 & 8 \\ 2 & 1 \\ 3 & 2 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \leq \begin{pmatrix} 12 \\ 3 \\ 4 \end{pmatrix} \text{ and } \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \geq \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

Dual linear program

$$\begin{aligned} \text{minimize: } & 12y_1 + 3y_2 + 4y_3 \\ \text{such that: } & 4y_1 + 2y_2 + 3y_3 \geq 2 \\ & 8y_1 + 1y_2 + 2y_3 \geq 3 \\ & y_1, y_2, y_3 \geq 0 \end{aligned}$$

Matrix form

Primal linear program

$$\begin{aligned} \text{maximize: } & 2x_1 + 3x_2 \\ \text{such that: } & 4x_1 + 8x_2 \leq 12 \\ & 2x_1 + 1x_2 \leq 3 \\ & 3x_1 + 2x_2 \leq 4 \\ & x_1, x_2 \geq 0 \end{aligned}$$

$$\max \begin{pmatrix} 2 & 3 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$

$$\text{with } \begin{pmatrix} 4 & 8 \\ 2 & 1 \\ 3 & 2 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \leq \begin{pmatrix} 12 \\ 3 \\ 4 \end{pmatrix} \text{ and } \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \geq \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

Dual linear program

$$\begin{aligned} \text{minimize: } & 12y_1 + 3y_2 + 4y_3 \\ \text{such that: } & 4y_1 + 2y_2 + 3y_3 \geq 2 \\ & 8y_1 + 1y_2 + 2y_3 \geq 3 \\ & y_1, y_2, y_3 \geq 0 \end{aligned}$$

$$\min \begin{pmatrix} 12 & 3 & 4 \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix}$$

$$\text{with } \begin{pmatrix} 4 & 2 & 3 \\ 8 & 1 & 2 \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix} \geq \begin{pmatrix} 2 \\ 3 \end{pmatrix} \text{ and } \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix} \geq \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

Matrix form

Primal linear program

$$\begin{aligned} \text{maximize: } & 2x_1 + 3x_2 \\ \text{such that: } & 4x_1 + 8x_2 \leq 12 \\ & 2x_1 + 1x_2 \leq 3 \\ & 3x_1 + 2x_2 \leq 4 \\ & x_1, x_2 \geq 0 \end{aligned}$$

$$\max \begin{pmatrix} 2 & 3 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$

$$\text{with } \begin{pmatrix} 4 & 8 \\ 2 & 1 \\ 3 & 2 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \leq \begin{pmatrix} 12 \\ 3 \\ 4 \end{pmatrix} \text{ and } \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \geq \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

Dual linear program

$$\begin{aligned} \text{minimize: } & 12y_1 + 3y_2 + 4y_3 \\ \text{such that: } & 4y_1 + 2y_2 + 3y_3 \geq 2 \\ & 8y_1 + 1y_2 + 2y_3 \geq 3 \\ & y_1, y_2, y_3 \geq 0 \end{aligned}$$

$$\min \begin{pmatrix} 12 & 3 & 4 \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix}$$

$$\text{with } \begin{pmatrix} 4 & 2 & 3 \\ 8 & 1 & 2 \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix} \geq \begin{pmatrix} 2 \\ 3 \end{pmatrix} \text{ and } \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix} \geq \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

Matrix form

Primal linear program

$$\begin{aligned} \text{maximize: } & 2x_1 + 3x_2 \\ \text{such that: } & 4x_1 + 8x_2 \leq 12 \\ & 2x_1 + 1x_2 \leq 3 \\ & 3x_1 + 2x_2 \leq 4 \\ & x_1, x_2 \geq 0 \end{aligned}$$

$$\max \begin{pmatrix} 2 & 3 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$

$$\text{with } \begin{pmatrix} 4 & 8 \\ 2 & 1 \\ 3 & 2 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \leq \begin{pmatrix} 12 \\ 3 \\ 4 \end{pmatrix} \text{ and } \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \geq \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

maximize $c^T x$ with $Ax \leq b$ and $x \geq 0$

Dual linear program

$$\begin{aligned} \text{minimize: } & 12y_1 + 3y_2 + 4y_3 \\ \text{such that: } & 4y_1 + 2y_2 + 3y_3 \geq 2 \\ & 8y_1 + 1y_2 + 2y_3 \geq 3 \\ & y_1, y_2, y_3 \geq 0 \end{aligned}$$

$$\min \begin{pmatrix} 12 & 3 & 4 \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix}$$

$$\text{with } \begin{pmatrix} 4 & 2 & 3 \\ 8 & 1 & 2 \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix} \geq \begin{pmatrix} 2 \\ 3 \end{pmatrix} \text{ and } \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix} \geq \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

minimize $b^T y$ with $A^T y \geq c$ and $y \geq 0$

Matrix form

Primal linear program

$$\begin{aligned} \text{maximize: } & 2x_1 + 3x_2 \\ \text{such that: } & 4x_1 + 8x_2 \leq 12 \\ & 2x_1 + 1x_2 \leq 3 \\ & 3x_1 + 2x_2 \leq 4 \\ & x_1, x_2 \geq 0 \end{aligned}$$

$$\max \begin{pmatrix} 2 & 3 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$

$$\text{with } \begin{pmatrix} 4 & 8 \\ 2 & 1 \\ 3 & 2 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \leq \begin{pmatrix} 12 \\ 3 \\ 4 \end{pmatrix} \text{ and } \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \geq \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

maximize $c^T x$ with $Ax \leq b$ and $x \geq 0$

Dual linear program

$$\begin{aligned} \text{minimize: } & 12y_1 + 3y_2 + 4y_3 \\ \text{such that: } & 4y_1 + 2y_2 + 3y_3 \geq 2 \\ & 8y_1 + 1y_2 + 2y_3 \geq 3 \\ & y_1, y_2, y_3 \geq 0 \end{aligned}$$

$$\min \begin{pmatrix} 12 & 3 & 4 \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix}$$

$$\text{with } \begin{pmatrix} 4 & 2 & 3 \\ 8 & 1 & 2 \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix} \geq \begin{pmatrix} 2 \\ 3 \end{pmatrix} \text{ and } \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix} \geq \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

minimize $b^T y$ with $A^T y \geq c$ and $y \geq 0$

Note: the dual of the dual is the primal

Duality theorem

Theorem (without proof)
For the linear programs

$$\text{maximize } c^T x \text{ with } Ax \leq b \text{ and } x \geq 0 \text{ and} \qquad \text{(P)}$$

$$\text{minimize } b^T y \text{ with } A^T y \geq c \text{ and } y \geq 0, \qquad \text{(D)}$$

exactly one of the following statements holds:

- Neither (P) nor (D) has a feasible solution.
- (P) is unbounded and (D) has no feasible solution or vice versa.
- Both have feasible solutions. Then the maximum of (P) equals the minimum of (D).

Duality theorem

Theorem (without proof)
For the linear programs

$$\text{maximize } c^T x \text{ with } Ax \leq b \text{ and } x \geq 0 \text{ and} \quad \text{(P)}$$

$$\text{minimize } b^T y \text{ with } A^T y \geq c \text{ and } y \geq 0, \quad \text{(D)}$$

exactly one of the following statements holds:

- Neither (P) nor (D) has a feasible solution.
- (P) is unbounded and (D) has no feasible solution or vice versa.
- Both have feasible solutions. Then the maximum of (P) equals the minimum of (D).

Notes

- the dual yields a perfect upper bound for the primal

Duality theorem

Theorem (without proof)
For the linear programs

$$\text{maximize } c^T x \text{ with } Ax \leq b \text{ and } x \geq 0 \text{ and} \quad \text{(P)}$$

$$\text{minimize } b^T y \text{ with } A^T y \geq c \text{ and } y \geq 0, \quad \text{(D)}$$

exactly one of the following statements holds:

- Neither (P) nor (D) has a feasible solution.
- (P) is unbounded and (D) has no feasible solution or vice versa.
- Both have feasible solutions. Then the maximum of (P) equals the minimum of (D).

Notes

- the dual yields a perfect upper bound for the primal
- works for arbitrary LPs; not just ones in the above form

Duality theorem

Theorem (without proof)
For the linear programs

$$\text{maximize } c^T x \text{ with } Ax \leq b \text{ and } x \geq 0 \text{ and} \qquad \text{(P)}$$

$$\text{minimize } b^T y \text{ with } A^T y \geq c \text{ and } y \geq 0, \qquad \text{(D)}$$

exactly one of the following statements holds:

- Neither (P) nor (D) has a feasible solution.
- (P) is unbounded and (D) has no feasible solution or vice versa.
- Both have feasible solutions. Then the maximum of (P) equals the minimum of (D).

Notes

- the dual yields a perfect upper bound for the primal
- works for arbitrary LPs; not just ones in the above form
- relation to vertex cover: the dual can be a nice tool to get a different perspective $\rightarrow$ exercise

Back to FPT: Lenstra's theorem

Theorem (without proof)
For $A \in \mathbb{Z}^{m \times n}$ and $b \in \mathbb{Z}^m$, one can answer the question whether there exists an $x \in \mathbb{N}^n$ with $Ax \leq b$ in $O(n^{2,5n}|A, b|)$ time, where $|A, b|$ is the length of the binary encoding of the instance.

Back to FPT: Lenstra's theorem

Theorem (without proof)
For $A \in \mathbb{Z}^{m \times n}$ and $b \in \mathbb{Z}^m$, one can answer the question whether there exists an $x \in \mathbb{N}^n$ with $Ax \leq b$ in $O(n^{2,5n}|A, b|)$ time, where $|A, b|$ is the length of the binary encoding of the instance.

It follows

(n is also called the *dimension* of the ILP)

- ILP (as decision problem) with parameter $n =$ (number of variables) is in FPT

Back to FPT: Lenstra's theorem

Theorem (without proof)
For $A \in \mathbb{Z}^{m \times n}$ and $b \in \mathbb{Z}^m$, one can answer the question whether there exists an $x \in \mathbb{N}^n$ with $Ax \leq b$ in $O(n^{2,5n}|A, b|)$ time, where $|A, b|$ is the length of the binary encoding of the instance.

It follows

(n is also called the *dimension* of the ILP)

- ILP (as decision problem) with parameter $n =$ (number of variables) is in FPT
- number of inequalities contributes only polynomially to the running time
- size of the numbers contributes only logarithmically to the running time

Back to FPT: Lenstra's theorem

Theorem (without proof)
For $A \in \mathbb{Z}^{m \times n}$ and $b \in \mathbb{Z}^m$, one can answer the question whether there exists an $x \in \mathbb{N}^n$ with $Ax \leq b$ in $O(n^{2,5n}|A, b|)$ time, where $|A, b|$ is the length of the binary encoding of the instance.

It follows

(n is also called the *dimension* of the ILP)

- ILP (as decision problem) with parameter $n =$ (number of variables) is in FPT
- number of inequalities contributes only polynomially to the running time
- size of the numbers contributes only logarithmically to the running time

Metatheorem
A parameterized problem with parameter k that can be formulated as an ILP with $f(k)$ many variables is in FPT.

Example: Closest String

Problem: Closest String (k is our parameter)
Given k strings $s_1, \ldots, s_k \in \Sigma^n$ and $d \in \mathbb{N}$. Is there a string $s \in \Sigma^n$, that has Hamming distance at most d for every s_i?

s_1 A B C F G D C G E B A
s_2 B B D A G D B G B E C
s_3 B A E F C A C G B B F

Example: Closest String

Problem: Closest String (k is our parameter)
Given k strings $s_1, \ldots, s_k \in \Sigma^n$ and $d \in \mathbb{N}$. Is there a string $s \in \Sigma^n$, that has Hamming distance at most d for every s_i?

s_1	A	B	C	F	G	D	C	G	E	B	A
s_2	B	B	D	A	G	D	B	G	B	E	C
s_3	B	A	E	F	C	A	C	G	B	B	F
s	B	B	E	F	G	D	C	G	B	B	C

Example: Closest String

Problem: Closest String (k is our parameter)
Given k strings $s_1, \ldots, s_k \in \Sigma^n$ and $d \in \mathbb{N}$. Is there a string $s \in \Sigma^n$, that has Hamming distance at most d for every s_i?

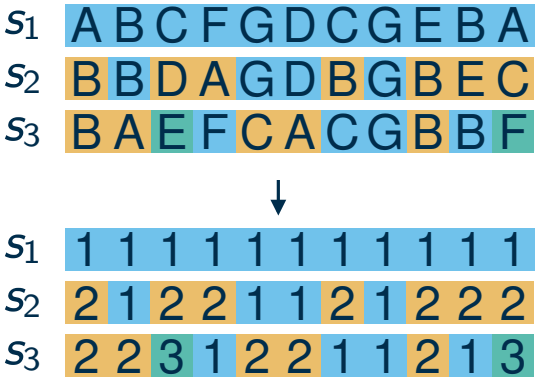
s_1	A	B	C	F	G	D	C	G	E	B	A
s_2	B	B	D	A	G	D	B	G	B	E	C
s_3	B	A	E	F	C	A	C	G	B	B	F

Observation

- only equality in each column relevant $\rightarrow$ equivalent input with $\Sigma = [k]$

Example: Closest String

Problem: Closest String (k is our parameter)
Given k strings $s_1, \ldots, s_k \in \Sigma^n$ and $d \in \mathbb{N}$. Is there a string $s \in \Sigma^n$, that has Hamming distance at most d for every s_i?



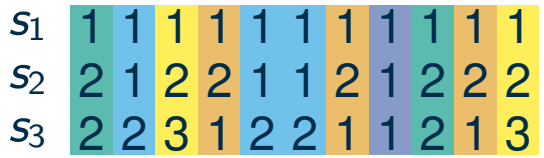
Observation

- only equality in each column relevant $\rightarrow$ equivalent input with $\Sigma = [k]$

Example: Closest String

Problem: Closest String (k is our parameter)
Given k strings $s_1, \ldots, s_k \in \Sigma^n$ and $d \in \mathbb{N}$. Is there a string $s \in \Sigma^n$, that has Hamming distance at most d for every s_i?

s_1 A B C F G D C G E B A
s_2 B B D A G D B G B E C
s_3 B A E F C A C G B B F



Observation

- only equality in each column relevant $\rightarrow$ equivalent input with $\Sigma = [k]$
- set of column types $\tau \rightarrow |\tau|$ only depends on k

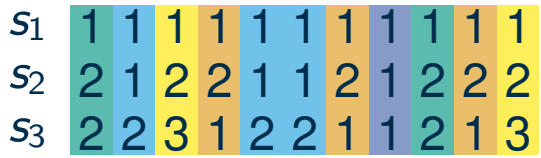
(bell number $B_k \leq k!$)

$$\tau = \left\{ \begin{matrix} 1 \\ 2 \\ 2 \end{matrix}, \begin{matrix} 1 \\ 1 \\ 2 \end{matrix}, \begin{matrix} 1 \\ 2 \\ 3 \end{matrix}, \begin{matrix} 1 \\ 2 \\ 1 \end{matrix}, \begin{matrix} 1 \\ 1 \\ 1 \end{matrix} \right\}$$

Example: Closest String

Problem: Closest String (k is our parameter)
Given k strings $s_1, \ldots, s_k \in \Sigma^n$ and $d \in \mathbb{N}$. Is there a string $s \in \Sigma^n$, that has Hamming distance at most d for every s_i?

s_1 A B C F G D C G E B A
s_2 B B D A G D B G B E C
s_3 B A E F C A C G B B F



Observation

- only equality in each column relevant $\rightarrow$ equivalent input with $\Sigma = [k]$
- set of column types $\tau \rightarrow |\tau|$ only depends on k
  (bell number $B_k \leq k!$)

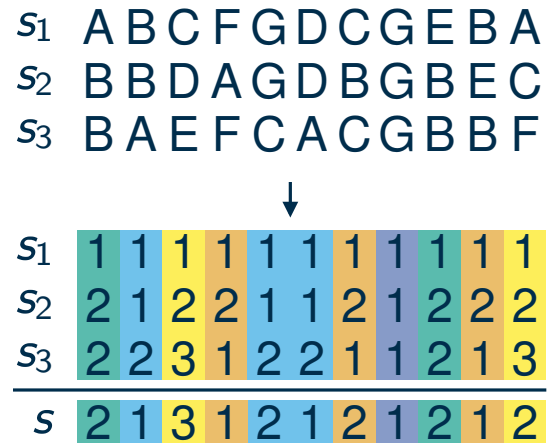
$$\tau = \left\{ \begin{matrix} 1 \\ 2 \\ 2 \end{matrix}, \begin{matrix} 1 \\ 1 \\ 2 \end{matrix}, \begin{matrix} 1 \\ 2 \\ 3 \end{matrix}, \begin{matrix} 1 \\ 2 \\ 1 \end{matrix}, \begin{matrix} 1 \\ 1 \\ 1 \end{matrix} \right\}$$

Formulation as ILP

- solution boils down to: How often is char $a \in \Sigma$ selected for column-type $t \in \tau$?

Example: Closest String

Problem: Closest String (k is our parameter)
Given k strings $s_1, \ldots, s_k \in \Sigma^n$ and $d \in \mathbb{N}$. Is there a string $s \in \Sigma^n$, that has Hamming distance at most d for every s_i?



Observation

- only equality in each column relevant $\rightarrow$ equivalent input with $\Sigma = [k]$
- set of column types $\tau \rightarrow |\tau|$ only depends on k
  (bell number $B_k \leq k!$)

$$\tau = \left\{ \begin{matrix} 1 \\ 2 \\ 2 \end{matrix}, \begin{matrix} 1 \\ 1 \\ 2 \end{matrix}, \begin{matrix} 1 \\ 2 \\ 3 \end{matrix}, \begin{matrix} 1 \\ 2 \\ 1 \end{matrix}, \begin{matrix} 1 \\ 1 \\ 1 \end{matrix} \right\}$$

Formulation as ILP

- solution boils down to: How often is char $a \in \Sigma$ selected for column-type $t \in \tau$?

Example: Closest String

Problem: Closest String (k is our parameter)
Given k strings $s_1, \ldots, s_k \in \Sigma^n$ and $d \in \mathbb{N}$. Is there a string $s \in \Sigma^n$, that has Hamming distance at most d for every s_i?

Observation

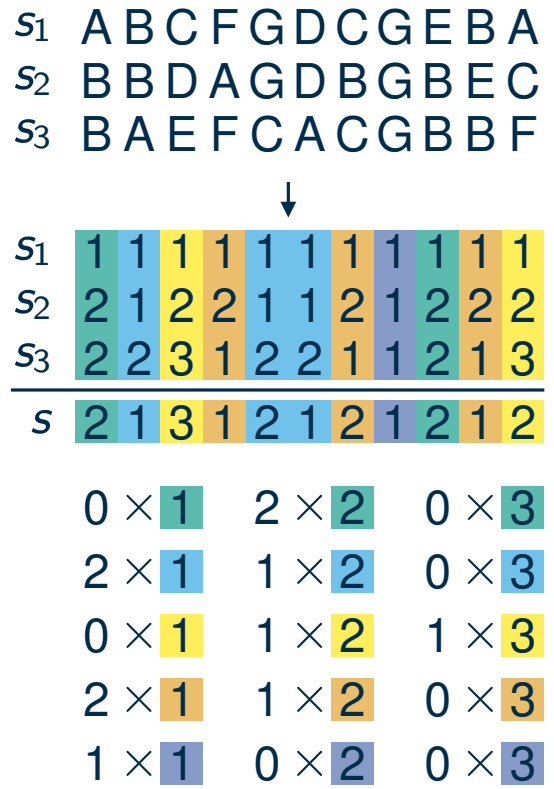
- only equality in each column relevant $\rightarrow$ equivalent input with $\Sigma = [k]$
- set of column types $\tau \rightarrow |\tau|$ only depends on k

(bell number $B_k \leq k!$)

$$\tau = \left\{ \begin{matrix} 1 \\ 2 \end{matrix}, \begin{matrix} 1 \\ 1 \end{matrix}, \begin{matrix} 1 \\ 2 \end{matrix}, \begin{matrix} 1 \\ 2 \end{matrix}, \begin{matrix} 1 \\ 1 \end{matrix} \right\}$$

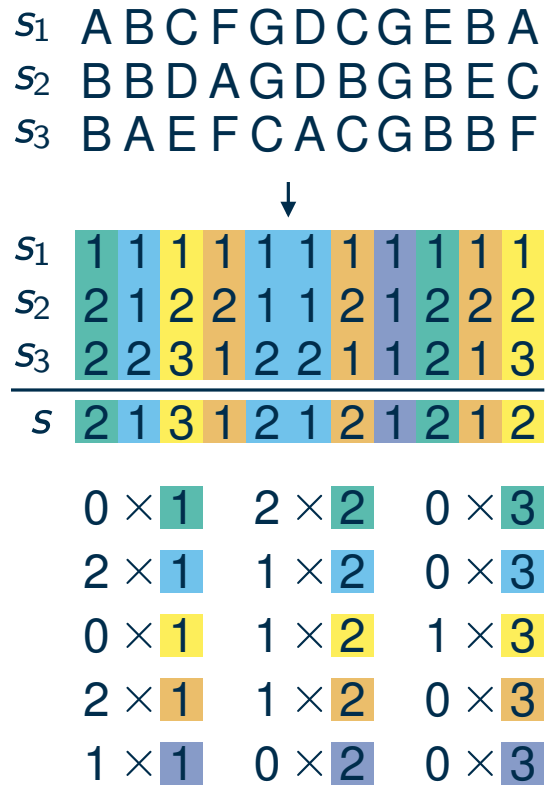
Formulation as ILP

- solution boils down to: How often is char $a \in \Sigma$ selected for column-type $t \in \tau$?



Example: Closest String

Problem: Closest String (k is our parameter)
Given k strings $s_1, \ldots, s_k \in \Sigma^n$ and $d \in \mathbb{N}$. Is there a string $s \in \Sigma^n$, that has Hamming distance at most d for every s_i?



Observation

- only equality in each column relevant $\rightarrow$ equivalent input with $\Sigma = [k]$
- set of column types $\tau \rightarrow |\tau|$ only depends on k
  (bell number $B_k \leq k!$)

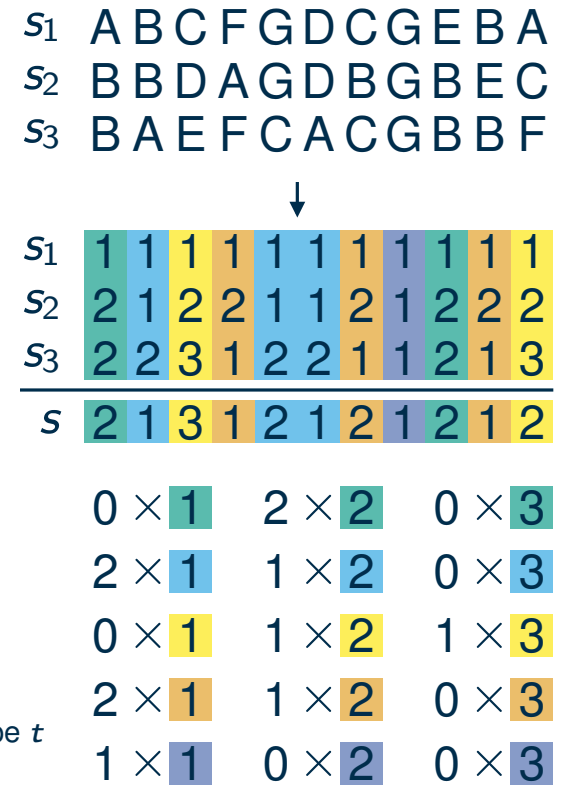
$$\tau = \left\{ \begin{matrix} 1 \\ 2 \end{matrix}, \begin{matrix} 1 \\ 1 \end{matrix}, \begin{matrix} 1 \\ 2 \end{matrix}, \begin{matrix} 1 \\ 2 \end{matrix}, \begin{matrix} 1 \\ 1 \end{matrix} \right\}$$

Formulation as ILP

- solution boils down to: How often is char $a \in \Sigma$ selected for column-type $t \in \tau$? $\rightarrow$ variable $x_{t,a}$

Example: Closest String

Problem: Closest String (k is our parameter)
Given k strings $s_1, \ldots, s_k \in \Sigma^n$ and $d \in \mathbb{N}$. Is there a string $s \in \Sigma^n$, that has Hamming distance at most d for every s_i?



Observation

- only equality in each column relevant $\rightarrow$ equivalent input with $\Sigma = [k]$
- set of column types $\tau \rightarrow |\tau|$ only depends on k
  (bell number $B_k \leq k!$)

$$\tau = \left\{ \begin{matrix} 1 \\ 2 \end{matrix}, \begin{matrix} 1 \\ 1 \end{matrix}, \begin{matrix} 1 \\ 2 \end{matrix}, \begin{matrix} 1 \\ 2 \end{matrix}, \begin{matrix} 1 \\ 1 \end{matrix} \right\}$$

Formulation as ILP

- solution boils down to: How often is char $a \in \Sigma$ selected for column-type $t \in \tau$? $\rightarrow$ variable $x_{t,a}$
- choose one character for each column

#chars selected for columns of type t — #columns of type t

$$\forall t \in \tau : \sum_{a \in \Sigma} x_{t,a} = \#t$$

Example: Closest String

Problem: Closest String (k is our parameter)
Given k strings $s_1, \ldots, s_k \in \Sigma^n$ and $d \in \mathbb{N}$. Is there a string $s \in \Sigma^n$, that has Hamming distance at most d for every s_i?

s_1 A B C F G D C G E B A
s_2 B B D A G D B G B E C
s_3 B A E F C A C G B B F

s_1	1	1	1	1	1	1	1	1	1	1	1
s_2	2	1	2	2	1	1	2	1	2	2	2
s_3	2	2	3	1	2	2	1	1	2	1	3
s	2	1	3	1	2	1	2	1	2	1	2

Observation

- only equality in each column relevant $\rightarrow$ equivalent input with $\Sigma = [k]$
- set of column types $\tau \rightarrow |\tau|$ only depends on k

(bell number $B_k \leq k!$)

$$\tau = \left\{ \begin{matrix} 1 \\ 2 \end{matrix}, \begin{matrix} 1 \\ 1 \end{matrix}, \begin{matrix} 1 \\ 2 \end{matrix}, \begin{matrix} 1 \\ 2 \end{matrix}, \begin{matrix} 1 \\ 1 \end{matrix} \right\}$$

0×1	2×2	0×3
2×1	1×2	0×3
0×1	1×2	1×3
2×1	1×2	0×3
1×1	0×2	0×3

Formulation as ILP

- solution boils down to: How often is char $a \in \Sigma$ selected for column-type $t \in \tau$? $\rightarrow$ variable $x_{t,a}$
- choose one character for each column
- distance $\leq d$ for every s_i

#chars selected for columns of type t — #columns of type t

$$\forall t \in \tau : \sum_{a \in \Sigma} x_{t,a} = \#t$$

$$\forall i \in [k] : \sum_{t \in \tau} \sum_{\substack{a \in \Sigma \\ a \neq t[i]}} x_{t,a} \leq d$$

#columns where selection disagrees with s_i

Example: Closest String

Problem: Closest String (k is our parameter)
Given k strings $s_1, \ldots, s_k \in \Sigma^n$ and $d \in \mathbb{N}$. Is there a string $s \in \Sigma^n$, that has Hamming distance at most d for every s_i?

s_1 A B C F G D C G E B A
s_2 B B D A G D B G B E C
s_3 B A E F C A C G B B F

s_1	1	1	1	1	1	1	1	1	1	1	1
s_2	2	1	2	2	1	1	2	1	2	2	2
s_3	2	2	3	1	2	2	1	1	2	1	3
s	2	1	3	1	2	1	2	1	2	1	2

Observation

- only equality in each column relevant $\rightarrow$ equivalent input with $\Sigma = [k]$
- set of column types $\tau \rightarrow |\tau|$ only depends on k
  (bell number $B_k \leq k!$)

$$\tau = \left\{ \begin{matrix} 1 \\ 2 \\ 2 \end{matrix}, \begin{matrix} 1 \\ 1 \\ 2 \end{matrix}, \begin{matrix} 1 \\ 2 \\ 3 \end{matrix}, \begin{matrix} 1 \\ 2 \\ 1 \end{matrix}, \begin{matrix} 1 \\ 1 \\ 1 \end{matrix} \right\}$$

0×1	2×2	0×3
2×1	1×2	0×3
0×1	1×2	1×3
2×1	1×2	0×3
1×1	0×2	0×3

Formulation as ILP

- solution boils down to: How often is char $a \in \Sigma$ selected for column-type $t \in \tau$? $\rightarrow$ variable $x_{t,a}$
- choose one character for each column
- distance $\leq d$ for every s_i
- yields ILP with "only" $k! \cdot k$ variables

#chars selected for columns of type t — #columns of type t

$$\forall t \in \tau : \sum_{a \in \Sigma} x_{t,a} = \#t$$

$$\forall i \in [k] : \sum_{t \in \tau} \sum_{\substack{a \in \Sigma \\ a \neq t[i]}} x_{t,a} \leq d$$

#columns where selection disagrees with s_i

Wrap-Up

Theorem
There is an FPT algorithm for Vertex Cover above LP with running time $4^k \cdot n^{O(1)}$.

Wrap-Up

Theorem
There is an FPT algorithm for VERTEX COVER ABOVE LP with running time $4^k \cdot n^{O(1)}$.

Other lower bounds

- the size of a maximum matching $\ell_M(G)$ is also a lower bound

Wrap-Up

Theorem
There is an FPT algorithm for VERTEX COVER ABOVE LP with running time $4^k \cdot n^{O(1)}$.

Other lower bounds

- the size of a maximum matching $\ell_M(G)$ is also a lower bound
- from duality it follows: $\ell_M(G) \leq \ell_{LP}(G)$ (see exercise sheet) $\Rightarrow$ today's algo is also FPT for VC ABOVE MATCHING

Wrap-Up

Theorem
There is an FPT algorithm for VERTEX COVER ABOVE LP with running time $4^k \cdot n^{O(1)}$.

Other lower bounds

- the size of a maximum matching $\ell_M(G)$ is also a lower bound
- from duality it follows: $\ell_M(G) \leq \ell_{LP}(G)$ (see exercise sheet) $\Rightarrow$ today's algo is also FPT for VC ABOVE MATCHING
- one can show: $2\ell_{LP}(G) - \ell_M(G)$ is also a lower bound (and it is stronger)

Wrap-Up

Theorem
There is an FPT algorithm for VERTEX COVER ABOVE LP with running time $4^k \cdot n^{O(1)}$.

Other lower bounds

- the size of a maximum matching $\ell_M(G)$ is also a lower bound
- from duality it follows: $\ell_M(G) \leq \ell_{LP}(G)$ (see exercise sheet) $\Rightarrow$ today's algo is also FPT for VC ABOVE MATCHING
- one can show: $2\ell_{LP}(G) - \ell_M(G)$ is also a lower bound (and it is stronger)
- VERTEX COVER ABOVE $2\ell_{LP}(G) - \ell_M(G)$ is also FPT

Wrap-Up

Theorem
There is an FPT algorithm for VERTEX COVER ABOVE LP with running time $4^k \cdot n^{O(1)}$.

Other lower bounds

- the size of a maximum matching $\ell_M(G)$ is also a lower bound
- from duality it follows: $\ell_M(G) \leq \ell_{LP}(G)$ (see exercise sheet) $\Rightarrow$ today's algo is also FPT for VC ABOVE MATCHING
- one can show: $2\ell_{LP}(G) - \ell_M(G)$ is also a lower bound (and it is stronger)
- VERTEX COVER ABOVE $2\ell_{LP}(G) - \ell_M(G)$ is also FPT

Duality

- systematic calculation of upper bounds
- strong duality theorem: very useful tool in many areas

Literature

Raising The Bar For Vertex Cover: Fixed-parameter Tractability Above A Higher Guarantee

- Shivam Garg, Geevarghese Philip [2016]
- contains the mentioned result for VERTEX COVER ABOVE $2\ell_{\mathrm{LP}}(G) - \ell_{\mathrm{M}}(G)$
- many additional references on the topic

doi.org/10.1137/1.9781611974331.ch80

Branch-and-Reduce Exponential/FPT Algorithms in Practice: A Case Study of Vertex Cover

- Takuya Akiba, Yoichi Iwata [2016]
- Branch-and-reduce for VERTEX COVER in practice

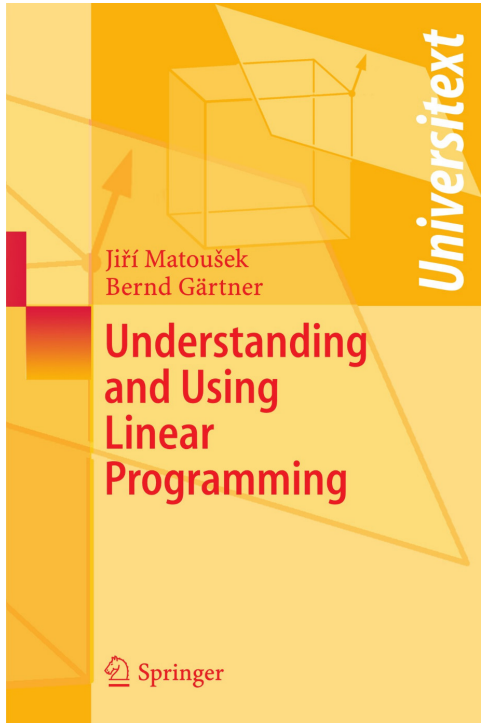
doi.org/10.1016/j.tcs.2015.09.023

WeGotYouCovered: The Winning Solver from the PACE 2019 Challenge, Vertex Cover Track

- Demian Hespe, Sebastian Lamm, Christian Schulz, Darren Strash [2020]
- fast algorithm for VERTEX COVER in practice

doi.org/10.1137/1.9781611976229.1

More literature



Understanding and Using Linear Programming

- exceptionally well written and compact book
- ~~free for KIT members~~

`link.springer.com/book/10.1007/978-3-540-30717-4`

Integer Programming in Parameterized Complexity: Three Miniatures

- Tomás Gavenciak, Dusan Knop, Martin Koutecký [2019]
- good overview about ILPs for parameterized problems; has many additional references

`drops.dagstuhl.de/opus/volltexte/2019/10222/`