Approximating the Sparsest k-Subgraph in Chordal Graphs

Rémi Watrigant, Marin Bougeret and Rodolphe Giroudeau

LIRMM, Montpellier, France

LIRMM
Laboratoire d'Informatique de Robotique et de Microélectronique de Montpellier

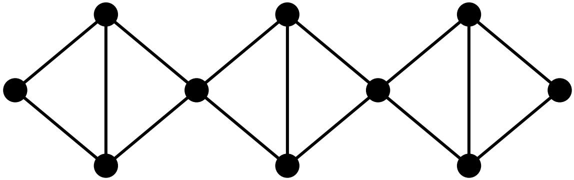
Workshop on Approximation and Online Algorithms
Sophia Antipolis, France
September 05-06, 2013

Sparsest k-Subgraph Problem (SkS)

Input: a simple undirected graph $G = (V, E)$, $k \leq |V|$.
Output: a set $S \subseteq V$ of size exactly k.
Goal: minimize $E(S)$ (the number of edges induced by S)

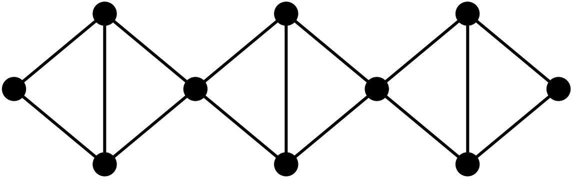
Sparsest k-Subgraph Problem (SkS)

Input: a simple undirected graph $G = (V, E)$, $k \leq |V|$.
Output: a set $S \subseteq V$ of size exactly k.
Goal: minimize $E(S)$ (the number of edges induced by S)



Sparsest k-Subgraph Problem (SkS)

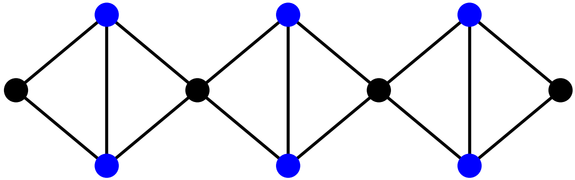
Input: a simple undirected graph $G = (V, E)$, $k \leq |V|$.
Output: a set $S \subseteq V$ of size exactly k.
Goal: minimize $E(S)$ (the number of edges induced by S)



$k = 6$: sparsest 6-subgraph =

Sparsest k-Subgraph Problem (SkS)

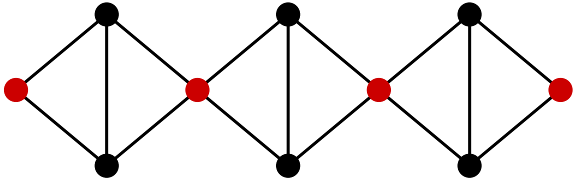
Input: a simple undirected graph $G = (V, E)$, $k \leq |V|$.
Output: a set $S \subseteq V$ of size exactly k.
Goal: minimize $E(S)$ (the number of edges induced by S)



$k = 6$: sparsest 6-subgraph = 3 edges

Sparsest k-Subgraph Problem (SkS)

Input: a simple undirected graph $G = (V, E)$, $k \leq |V|$.
Output: a set $S \subseteq V$ of size exactly k.
Goal: minimize $E(S)$ (the number of edges induced by S)



$k = 6$: sparsest 6-subgraph = 3 edges

$k = 4$: sparsest 4-subgraph = 0 edges

Sparsest k-Subgraph Problem (SkS)

Input: a simple undirected graph $G = (V, E)$, $k \leq |V|$.
Output: a set $S \subseteq V$ of size exactly k.
Goal: minimize $E(S)$ (the number of edges induced by S)

- Generalization of INDEPENDENT SET
  $\Rightarrow$ SkS NP-hard in general graphs (+ inapproximable)

Sparsest k-Subgraph Problem (SkS)

Input: a simple undirected graph $G = (V, E)$, $k \leq |V|$.
Output: a set $S \subseteq V$ of size exactly k.
Goal: minimize $E(S)$ (the number of edges induced by S)

- Generalization of INDEPENDENT SET
  $\Rightarrow$ SkS *NP*-hard in general graphs (+ inapproximable)

- Related problems:

Sparsest k-Subgraph Problem (SkS)

Input: a simple undirected graph $G = (V, E)$, $k \leq |V|$.
Output: a set $S \subseteq V$ of size exactly k.
Goal: minimize $E(S)$ (the number of edges induced by S)

- Generalization of INDEPENDENT SET
  $\Rightarrow$ SkS NP-hard in general graphs (+ inapproximable)

- Related problems:
  - maximisation version: Densest k-Subgraph (DkS)

exact result for DkS on the class $\mathcal{C}$

$$\Updownarrow$$

exact result for SkS on the class $\bar{\mathcal{C}}$

Sparsest k-Subgraph Problem (SkS)

Input: a simple undirected graph $G = (V, E)$, $k \leq |V|$.
Output: a set $S \subseteq V$ of size exactly k.
Goal: minimize $E(S)$ (the number of edges induced by S)

- Generalization of INDEPENDENT SET
  $\Rightarrow$ SkS NP-hard in general graphs (+ inapproximable)

- Related problems:
  - maximisation version: Densest k-Subgraph (DkS)

    exact result for DkS on the class $\mathcal{C}$
    $\Updownarrow$
    exact result for SkS on the class $\bar{\mathcal{C}}$

  - dual version: Maximum Vertex Coverage (MVC)

    k vertices covering the max. number of edges
    $\Updownarrow$
    $(n-k)$ vertices inducing the min. number of edges

Sparsest k-Subgraph Problem (SkS)

Input: a simple undirected graph $G = (V, E)$, $k \leq |V|$.
Output: a set $S \subseteq V$ of size exactly k.
Goal: minimize $E(S)$ (the number of edges induced by S)

- Generalization of INDEPENDENT SET
  $\Rightarrow$ SkS NP-hard in general graphs (+ inapproximable)

- Related problems:
  - maximisation version: Densest k-Subgraph (DkS)

    exact result for DkS on the class $\mathcal{C}$

    $\Updownarrow$

    exact result for SkS on the class $\bar{\mathcal{C}}$

  - dual version: Maximum Vertex Coverage (MVC)

    k vertices covering the max. number of edges

    $\Updownarrow$

    $(n-k)$ vertices inducing the min. number of edges

but approximation do not transfer...

Summary:

Densest k-Subgraph	Sparsest k-Subgraph

Summary:

Densest k-Subgraph

- General graphs:
  NP-hard, no PTAS [Khot,'04], APX = OPEN
  $O(n^d)$ approx. for some $d \leq 1/3$ [Feige,'01]

Sparsest k-Subgraph

- General graphs:
  NP-hard (from Independent Set)
  no approx. (unbounded ratio)

Summary:

Densest k-Subgraph

- General graphs:
  NP-hard, no PTAS [Khot,'04], APX = OPEN
  $O(n^d)$ approx. for some $d \leq 1/3$ [Feige,'01]
- Perfect graphs:
  NP-hard [Corneil,Perl,'84]

Sparsest k-Subgraph

- General graphs:
  NP-hard (from Independent Set)
  no approx. (unbounded ratio)
- Perfect graphs:
  NP-hard [Corneil,Perl,'84]

Summary:

Densest k-Subgraph

- General graphs:
  NP-hard, no PTAS [Khot,'04], APX = OPEN
  $O(n^d)$ approx. for some $d \leq 1/3$ [Feige,'01]
- Perfect graphs:
  NP-hard [Corneil,Perl,'84]
- Chordal graphs:
  NP-hard [Corneil,Perl,'84]
  3-approx. [Liazi,Milis,Zissimopoulos,'08]

Sparsest k-Subgraph

- General graphs:
  NP-hard (from Independent Set)
  no approx. (unbounded ratio)
- Perfect graphs:
  NP-hard [Corneil,Perl,'84]
- Chordal graphs:
  NP-hard [Bougeret,Giroudeau,W.,'13]
  2-approx. [Bougeret,Giroudeau,W.,'13]

Summary:

Densest k-Subgraph

- General graphs:
  NP-hard, no PTAS [Khot,'04], APX = OPEN
  $O(n^d)$ approx. for some $d \leq 1/3$ [Feige,'01]
- Perfect graphs:
  NP-hard [Corneil,Perl,'84]
- Chordal graphs:
  NP-hard [Corneil,Perl,'84]
  3-approx. [Liazi,Milis,Zissimopoulos,'08]
- Interval graphs:
  NP-h/Poly : OPEN
  $PTAS$ [Nonner,'11]

Sparsest k-Subgraph

- General graphs:
  NP-hard (from Independent Set)
  no approx. (unbounded ratio)
- Perfect graphs:
  NP-hard [Corneil,Perl,'84]
- Chordal graphs:
  NP-hard [Bougeret,Giroudeau,W.,'13]
  2-approx. [Bougeret,Giroudeau,W.,'13]
- Interval graphs:
  NP-h/Poly : OPEN

Summary:

Densest k-Subgraph

- General graphs:
  NP-hard, no PTAS [Khot,'04], APX = OPEN
  $O(n^d)$ approx. for some $d \leq 1/3$ [Feige,'01]
- Perfect graphs:
  NP-hard [Corneil,Perl,'84]
- Chordal graphs:
  NP-hard [Corneil,Perl,'84]
  3-approx. [Liazi,Milis,Zissimopoulos,'08]
- Interval graphs:
  NP-h/Poly : OPEN
  $PTAS$ [Nonner,'11]
- Proper interval graphs:
  NP-h/Poly : OPEN

Sparsest k-Subgraph

- General graphs:
  NP-hard (from Independent Set)
  no approx. (unbounded ratio)
- Perfect graphs:
  NP-hard [Corneil,Perl,'84]
- Chordal graphs:
  NP-hard [Bougeret,Giroudeau,W.,'13]
  2-approx. [Bougeret,Giroudeau,W.,'13]
- Interval graphs:
  NP-h/Poly : OPEN
- Proper interval graphs:
  NP-h/Poly : OPEN
  $PTAS$ [Bougeret,Giroudeau,W.,'13]

Summary:

Densest k-Subgraph

- General graphs:
  NP-hard, no PTAS [Khot,'04], APX = OPEN
  $O(n^d)$ approx. for some $d \leq 1/3$ [Feige,'01]
- Perfect graphs:
  NP-hard [Corneil,Perl,'84]
- Chordal graphs:
  NP-hard [Corneil,Perl,'84]
  3-approx. [Liazi,Milis,Zissimopoulos,'08]
- Interval graphs:
  NP-h/Poly : OPEN
  $PTAS$ [Nonner,'11]
- Proper interval graphs:
  NP-h/Poly : OPEN
- Bipartite graphs:
  NP-h [Corneil,Perl,1984]

Sparsest k-Subgraph

- General graphs:
  NP-hard (from Independent Set)
  no approx. (unbounded ratio)
- Perfect graphs:
  NP-hard [Corneil,Perl,'84]
- Chordal graphs:
  NP-hard [Bougeret,Giroudeau,W.,'13]
  2-approx. [Bougeret,Giroudeau,W.,'13]
- Interval graphs:
  NP-h/Poly : OPEN
- Proper interval graphs:
  NP-h/Poly : OPEN
  $PTAS$ [Bougeret,Giroudeau,W.,'13]
- Bipartite graphs:
  NP-h [Joret et al.,'13][Apollonio et al.,'13]

Summary:

Densest k-Subgraph

- General graphs:
  NP-hard, no PTAS [Khot,'04], APX = OPEN
  $O(n^d)$ approx. for some $d \leq 1/3$ [Feige,'01]
- Perfect graphs:
  NP-hard [Corneil,Perl,'84]
- Chordal graphs:
  NP-hard [Corneil,Perl,'84]
  3-approx. [Liazi,Milis,Zissimopoulos,'08]
- Interval graphs:
  NP-h/Poly : OPEN
  $PTAS$ [Nonner,'11]
- Proper interval graphs:
  NP-h/Poly : OPEN
- Bipartite graphs:
  NP-h [Corneil,Perl,1984]
- Planar graphs:
  NP-h/Poly : OPEN

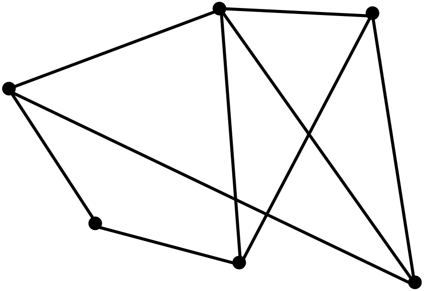
Sparsest k-Subgraph

- General graphs:
  NP-hard (from Independent Set)
  no approx. (unbounded ratio)
- Perfect graphs:
  NP-hard [Corneil,Perl,'84]
- Chordal graphs:
  NP-hard [Bougeret,Giroudeau,W.,'13]
  2-approx. [Bougeret,Giroudeau,W.,'13]
- Interval graphs:
  NP-h/Poly : OPEN
- Proper interval graphs:
  NP-h/Poly : OPEN
  $PTAS$ [Bougeret,Giroudeau,W.,'13]
- Bipartite graphs:
  NP-h [Joret et al.,'13][Apollonio et al.,'13]
- Planar graphs:
  NP-h (from Independent Set)

Chordal Graphs

Definition

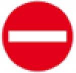
A graph G is chordal if it does not contain any cycle of length four or more as an induced subgraph.

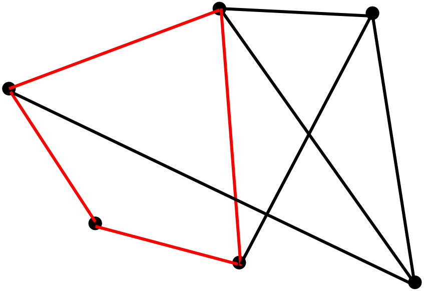


Chordal Graphs

Definition

A graph G is chordal if it does not contain any cycle of length four or more as an induced subgraph.

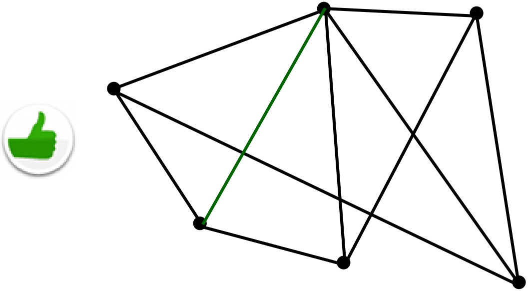




Chordal Graphs

Definition

A graph G is chordal if it does not contain any cycle of length four or more as an induced subgraph.



Chordal Graphs

Definition

A graph G is chordal if it does not contain any cycle of length four or more as an induced subgraph.

Folklore

A graph is chordal iff it admits a simplicial elimination order.

$$v_1 \quad .... \quad v_i \quad .... \quad v_j \quad .... \quad v_k \quad .... \quad v_l \quad .... \quad v_n$$

Chordal Graphs

Definition

A graph G is chordal if it does not contain any cycle of length four or more as an induced subgraph.

Folklore

A graph is chordal iff it admits a simplicial elimination order.

$$v_1 \quad .... \quad v_i \quad .... \quad v_j \quad .... \quad v_k \quad .... \quad v_l \quad .... \quad v_n$$

For all $i \in \{1, ..., n\}$, the neighbourhood of v_i in $G[v_i, ..., v_n]$ is a clique

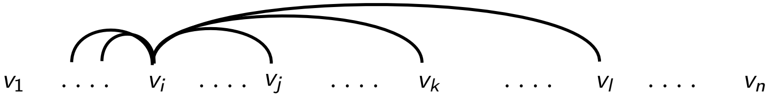
Chordal Graphs

Definition

A graph G is chordal if it does not contain any cycle of length four or more as an induced subgraph.

Folklore

A graph is chordal iff it admits a simplicial elimination order.



For all $i \in \{1, ..., n\}$, the neighbourhood of v_i in $G[v_i, ..., v_n]$ is a clique

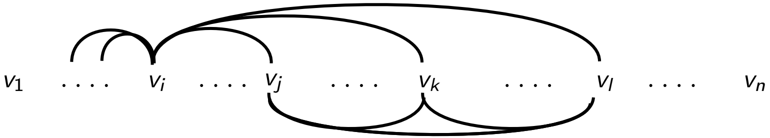
Chordal Graphs

Definition

A graph G is chordal if it does not contain any cycle of length four or more as an induced subgraph.

Folklore

A graph is chordal iff it admits a simplicial elimination order.



For all $i \in \{1, ..., n\}$, the neighbourhood of v_i in $G[v_i, ..., v_n]$ is a clique

Chordal Graphs

Definition

A graph G is chordal if it does not contain any cycle of length four or more as an induced subgraph.

Folklore

A graph is chordal iff it admits a simplicial elimination order.
Such an ordering can be found in polynomial time.

2-approximation in chordal graphs

Idea of the algorithm:

2-approximation in chordal graphs

Idea of the algorithm:

- **1.** first pick a maximum independent set

2-approximation in chordal graphs

Idea of the algorithm:

- **1.** first pick a maximum independent set
- **2.** pick another maximum independent set... etc. until we have k vertices

2-approximation in chordal graphs

Idea of the algorithm:

- **1.** first pick a maximum independent set
- ~~**2.** pick another maximum independent set... etc. until we have k vertices~~

2-approximation in chordal graphs

Idea of the algorithm:

- **1.** first pick a maximum independent set
- ~~**2.** pick another maximum independent set... etc. until we have *k* vertices~~
- **2.** put weights on remaining vertices (neighborhood in the partial solution)

2-approximation in chordal graphs

Idea of the algorithm:

- **1.** first pick a maximum independent set
- ~~**2.** pick another maximum independent set... etc. until we have k vertices~~
- **2.** put weights on remaining vertices (neighborhood in the partial solution)
- **3.** pick a maximum independent set among vertices of minimum weight
- **4.** go back to **2.** until we have k vertices

2-approximation in chordal graphs

Idea of the algorithm:

- **1.** first pick a maximum independent set
- ~~**2.** pick another maximum independent set... etc. until we have k vertices~~
- **2.** put weights on remaining vertices (neighborhood in the partial solution)
- **3.** pick a maximum independent set among vertices of minimum weight
- **4.** go back to **2.** until we have k vertices

Output of the algorithm:

- sequence of layers $L_0, \cdots, L_t$
- each layer L_i is an independent set of vertices of weight i

(having i neighbors in the partial solution)

Idea of the analysis:

2-approximation in chordal graphs

Idea of the algorithm:

- **1.** first pick a maximum independent set
- ~~**2.** pick another maximum independent set... etc. until we have k vertices~~
- **2.** put weights on remaining vertices (neighborhood in the partial solution)
- **3.** pick a maximum independent set among vertices of minimum weight
- **4.** go back to **2.** until we have k vertices

Output of the algorithm:

- sequence of layers $L_0, \cdots, L_t$
- each layer L_i is an independent set of vertices of weight i

(having i neighbors in the partial solution)

Idea of the analysis:

- restructuration of an optimal solution
  OPT$= OPT_0 \rightarrow OPT_1 \rightarrow ... \rightarrow OPT_{i-1} \rightarrow OPT_i \rightarrow ... \rightarrow OPT_t =$ **ALGO**
- how: layer by layer

2-approximation in chordal graphs

Idea of the algorithm:

- **1.** first pick a maximum independent set
- ~~**2.** pick another maximum independent set... etc. until we have k vertices~~
- **2.** put weights on remaining vertices (neighborhood in the partial solution)
- **3.** pick a maximum independent set among vertices of minimum weight
- **4.** go back to **2.** until we have k vertices

Output of the algorithm:

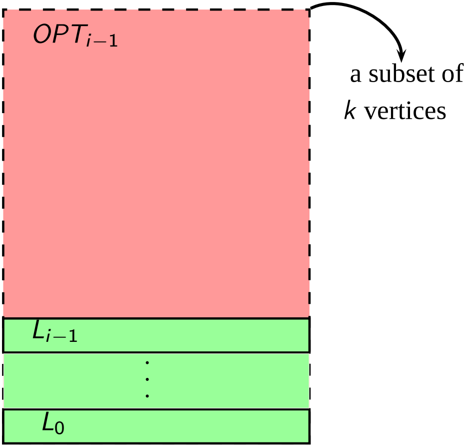
- sequence of layers $L_0, \cdots, L_t$
- each layer L_i is an independent set of vertices of weight i

  (having i neighbors in the partial solution)

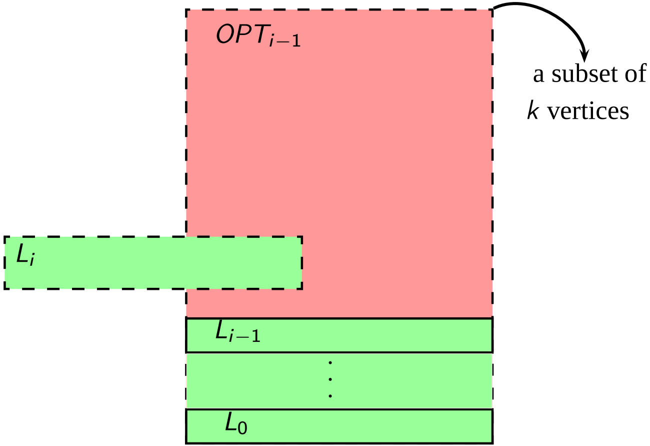
Idea of the analysis:

- restructuration of an optimal solution
  OPT$= OPT_0 \rightarrow OPT_1 \rightarrow \ldots \rightarrow OPT_{i-1} \rightarrow OPT_i \rightarrow \ldots \rightarrow OPT_t =$ **ALGO**
- how: layer by layer
- restructuration $\mathbf{OPT}_{i-1} \rightarrow \mathbf{OPT}_i$:
  suppose that layers $L_0, \ldots, L_{i-1}$ are the same for $\mathbf{OPT}_{i-1}$ and **ALGO**

2-approximation in chordal graphs: idea of restructuration

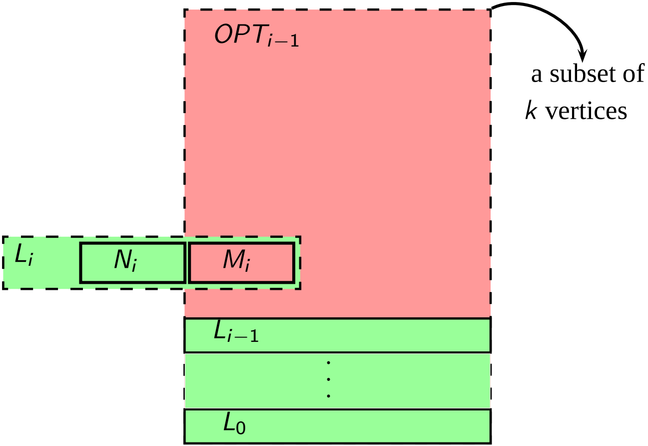


2-approximation in chordal graphs: idea of restructuration



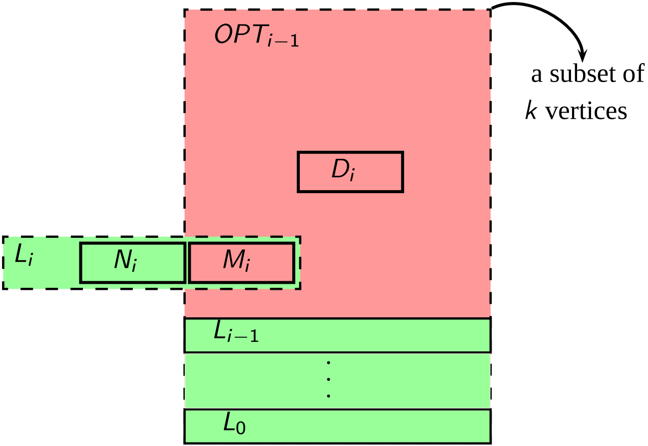
- idea : to "bring" the layer L_i into OPT_{i-1}

2-approximation in chordal graphs: idea of restructuration



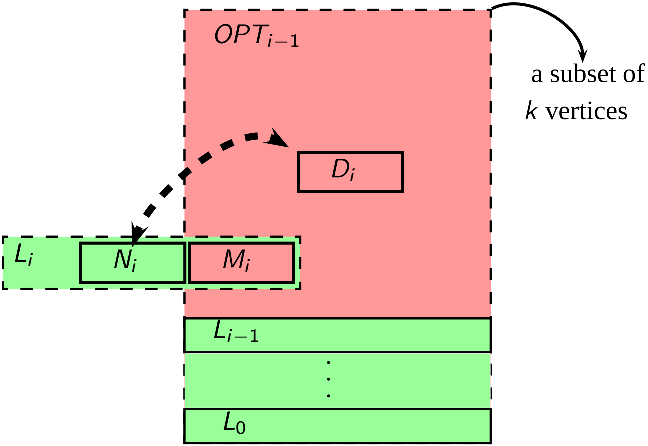
- idea : to "bring" the layer L_i into OPT_{i-1}
- $L_i = N_i \cup M_i$, with $M_i = L_i \cap OPT_{i-1}$

2-approximation in chordal graphs: idea of restructuration



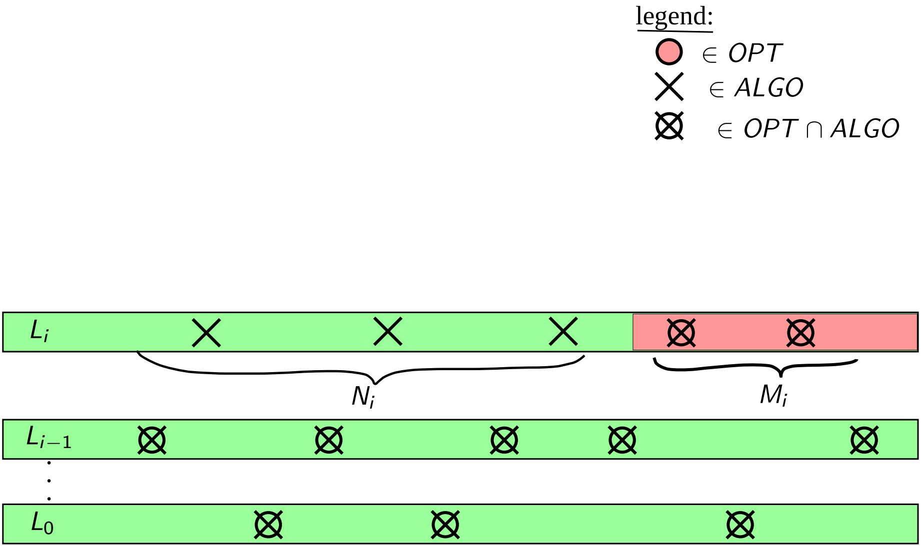
- idea : to "bring" the layer L_i into OPT_{i-1}
- $L_i = N_i \cup M_i$, with $M_i = L_i \cap OPT_{i-1}$
- goal: to find a set $D_i \subseteq OPT_{i-1}$

2-approximation in chordal graphs: idea of restructuration

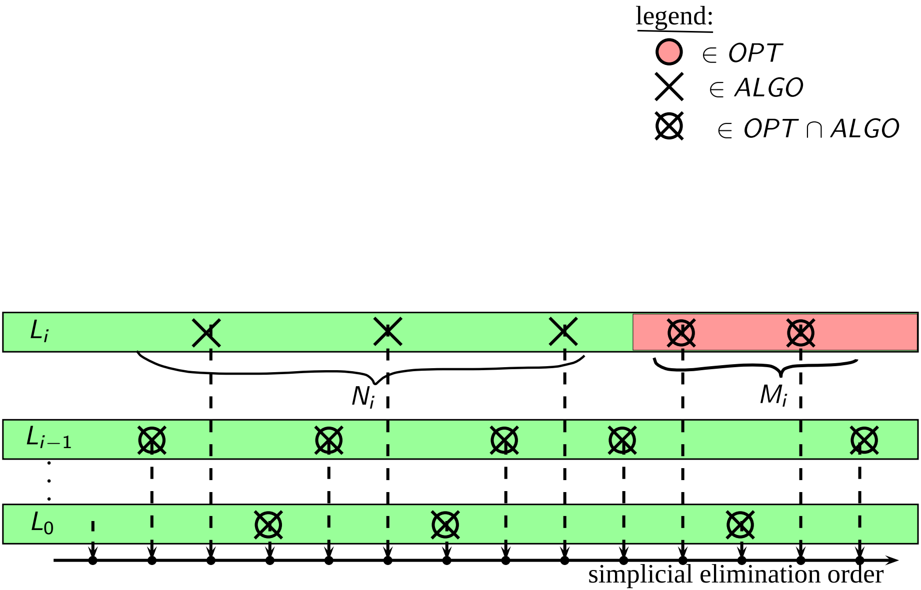


- idea : to "bring" the layer L_i into OPT_{i-1}
- $L_i = N_i \cup M_i$, with $M_i = L_i \cap OPT_{i-1}$
- goal: to find a set $D_i \subseteq OPT_{i-1}$
- replace D_i by N_i

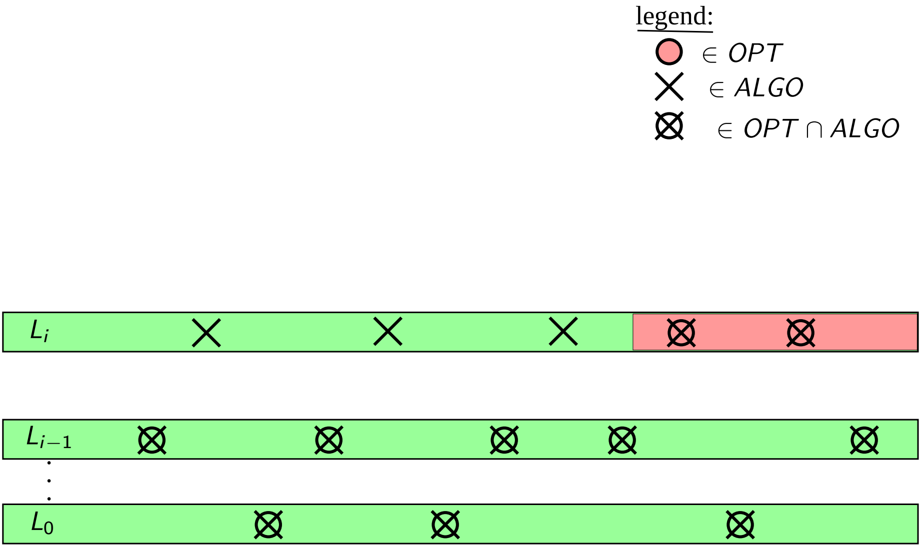
2-approximation in chordal graphs



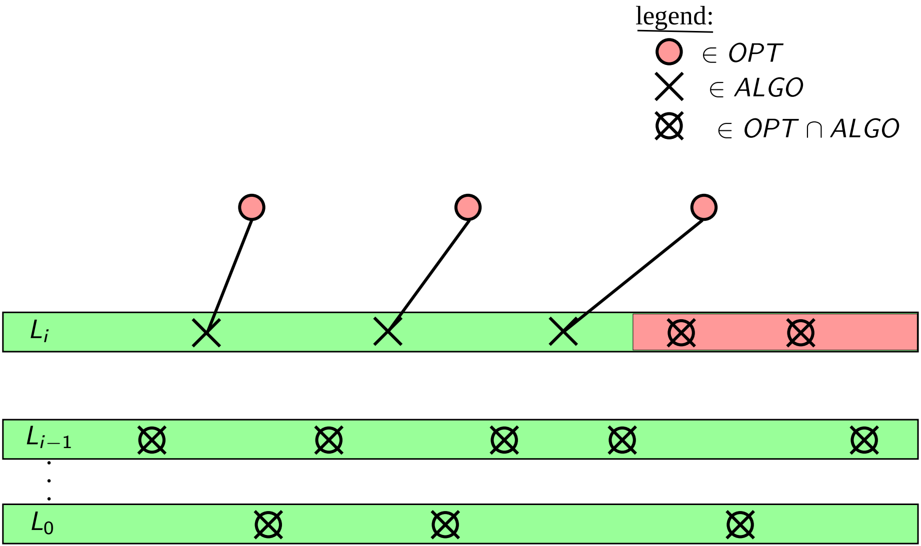
2-approximation in chordal graphs



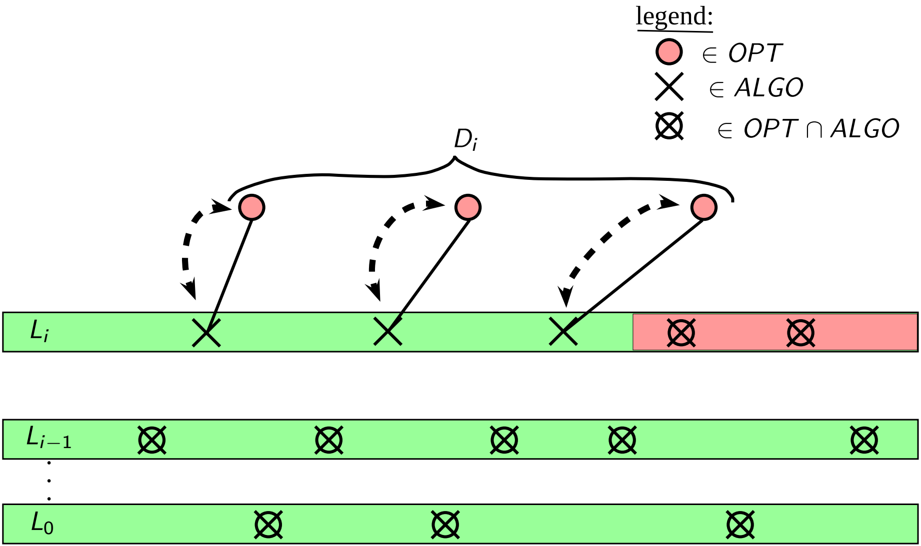
2-approximation in chordal graphs



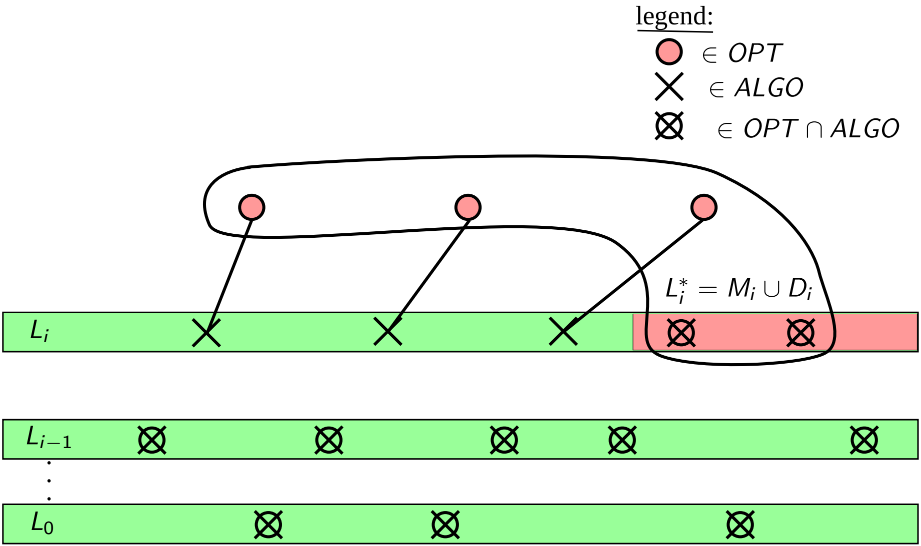
2-approximation in chordal graphs



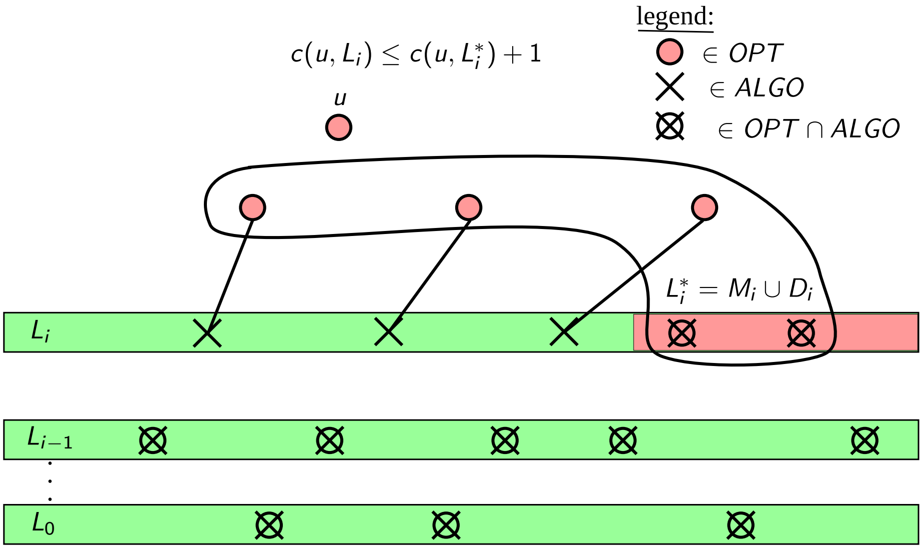
2-approximation in chordal graphs



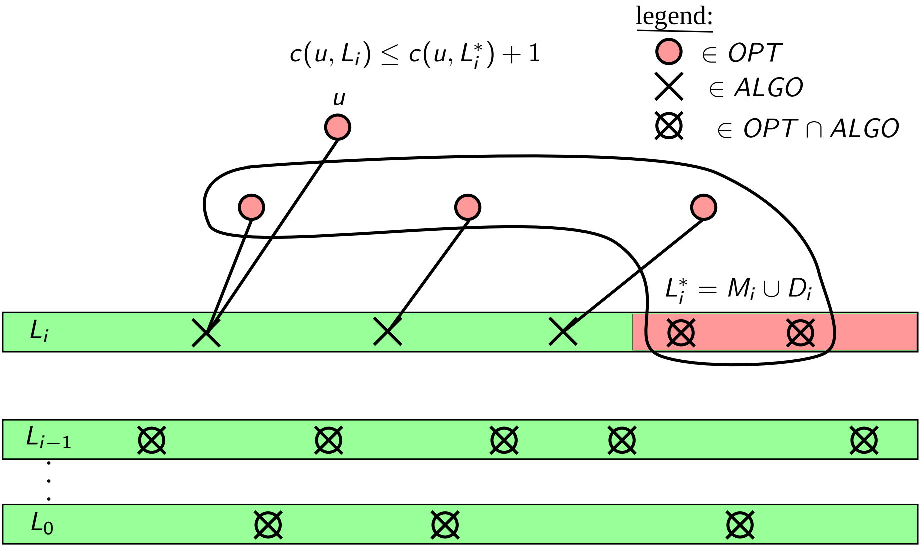
2-approximation in chordal graphs



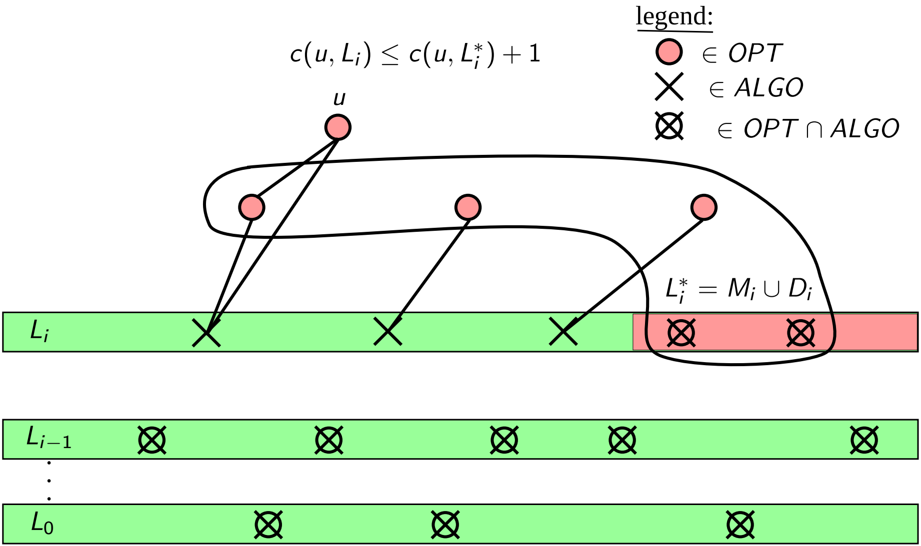
2-approximation in chordal graphs



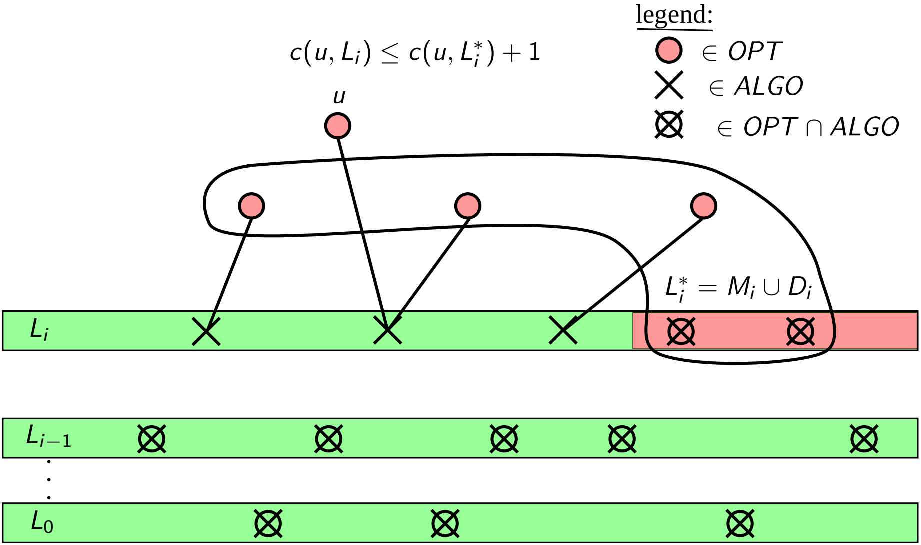
2-approximation in chordal graphs



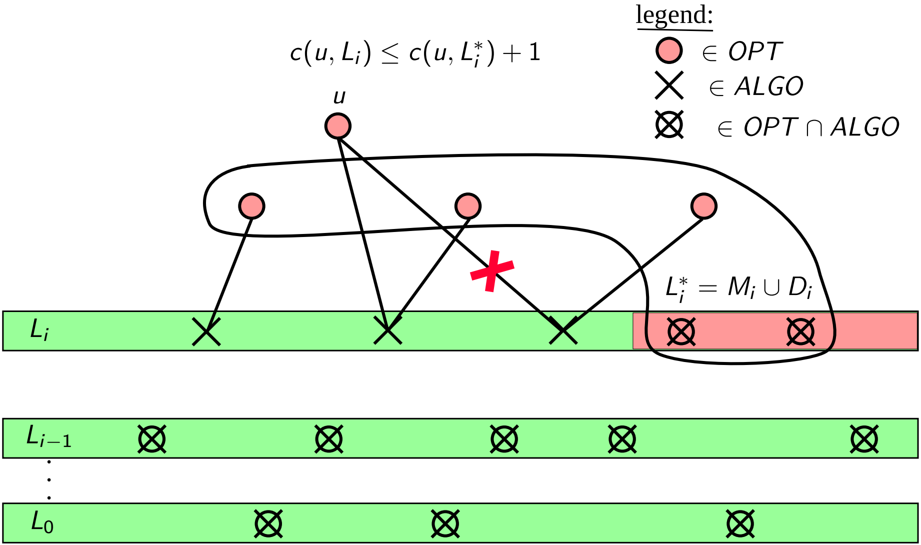
2-approximation in chordal graphs

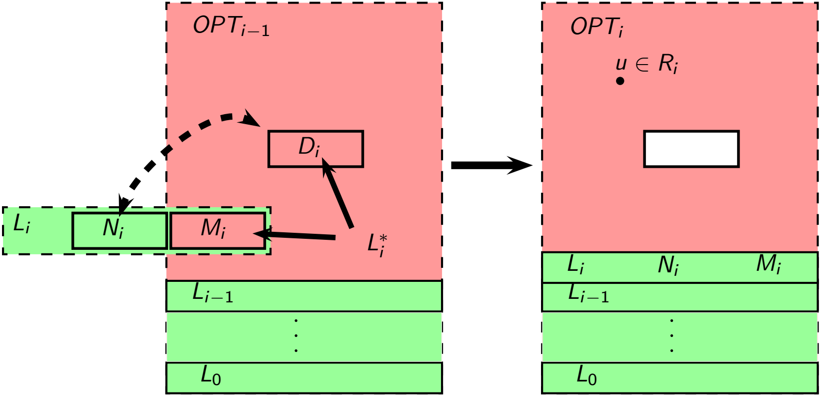


2-approximation in chordal graphs

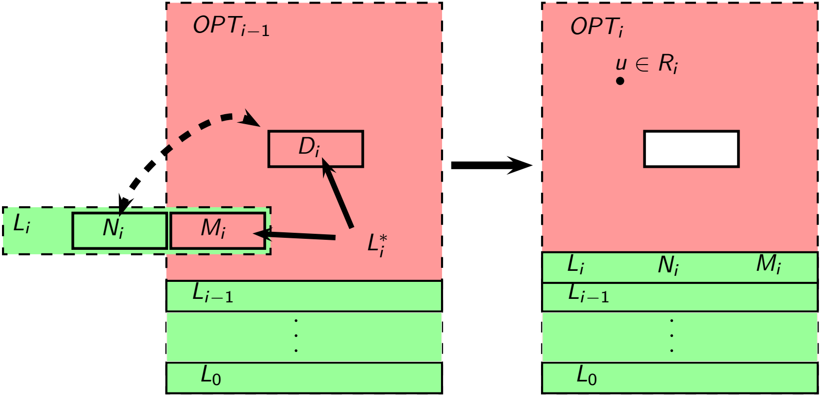


2-approximation in chordal graphs

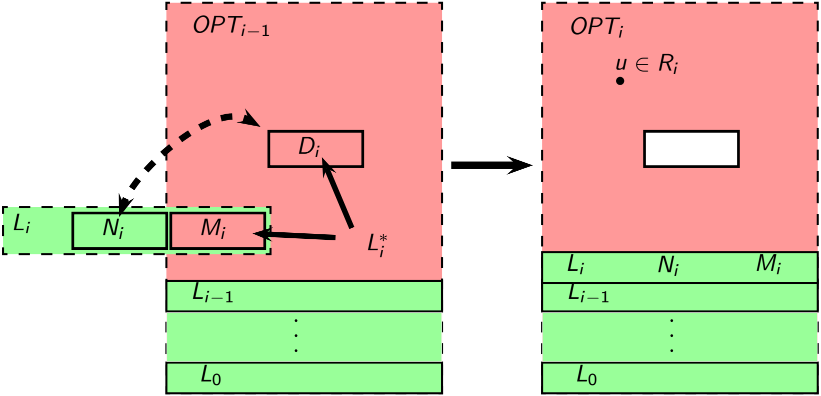




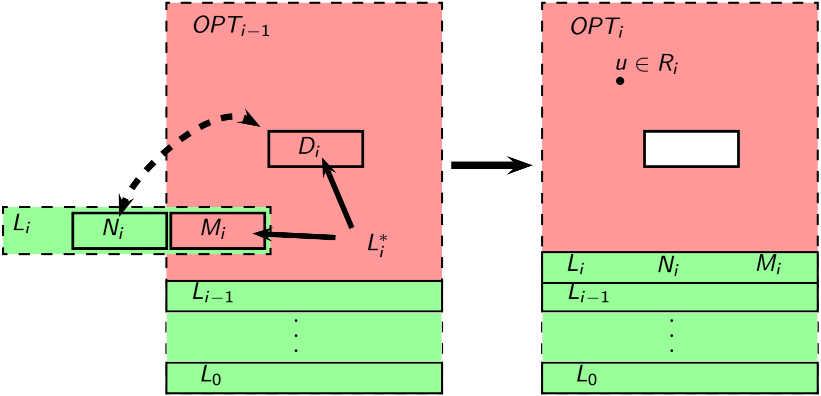
$\forall u \in R_i$ we have $c(u, L_i) \leq c(u, L_i^*) + 1$



$\forall u \in R_i$ we have $c(u, L_i) \leq c(u, L_i^*) + 1 \leq_{hyp} 2 \cdot c(u, L_i^*)$

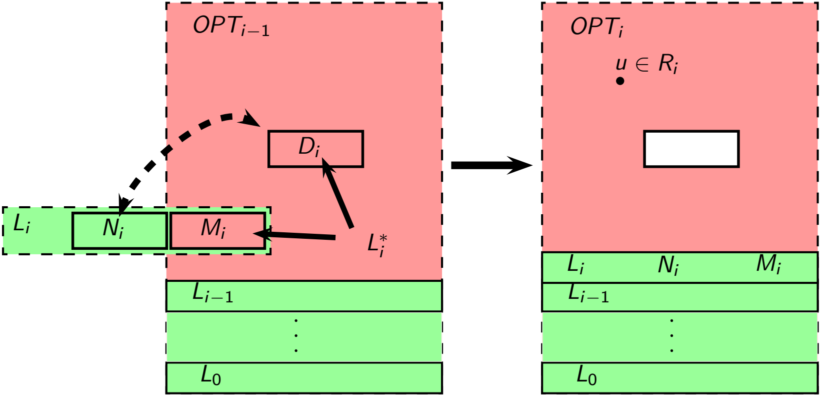


$\forall u \in R_i$ we have $c(u, L_i) \leq c(u, L_i^*) + 1 \leq_{hyp} 2 \cdot c(u, L_i^*) \Rightarrow c(R_i, L_i) \leq 2 \cdot c(u, L_i^*)$



$\forall u \in R_i$ we have $c(u, L_i) \leq c(u, L_i^*) + 1 \leq_{hyp} 2 \cdot c(u, L_i^*) \Rightarrow c(R_i, L_i) \leq 2 \cdot c(u, L_i^*)$
Then we can show:

$$c(OPT_i) \leq c(OPT_{i-1}) + c(R_i, L_i^*)$$

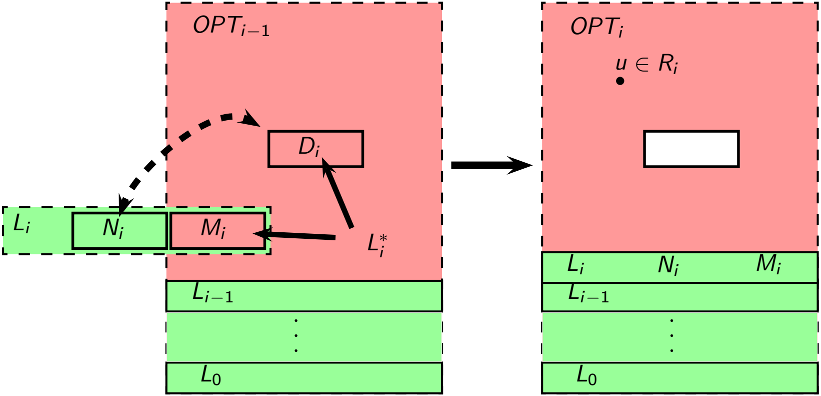


$\forall u \in R_i$ we have $c(u, L_i) \leq c(u, L_i^*) + 1 \leq_{hyp} 2 \cdot c(u, L_i^*) \Rightarrow c(R_i, L_i) \leq 2 \cdot c(u, L_i^*)$
Then we can show:

$$c(OPT_i) \leq c(OPT_{i-1}) + c(R_i, L_i^*)$$

Thus:

$$c(OPT_t) \leq c(OPT_0) + \sum_{i=1}^{t} c(R_i, L_i^*)$$



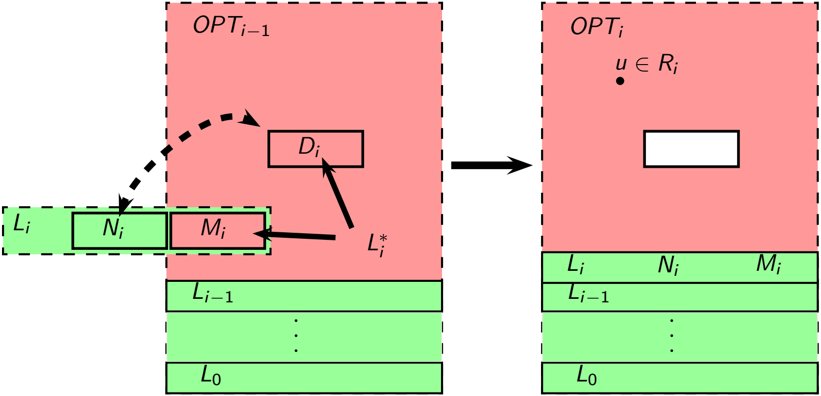
$\forall u \in R_i$ we have $c(u, L_i) \leq c(u, L_i^*) + 1 \leq_{hyp} 2 \cdot c(u, L_i^*) \Rightarrow c(R_i, L_i) \leq 2 \cdot c(u, L_i^*)$
Then we can show:

$$c(OPT_i) \leq c(OPT_{i-1}) + c(R_i, L_i^*)$$

Thus:

$$c(OPT_t) \leq c(OPT_0) + \sum_{i=1}^{t} c(R_i, L_i^*)$$

$$\Leftrightarrow c(ALGO) \leq c(OPT) + c(OPT)$$



Theorem:

There is a (tight) 2-approximation for Sparsest k-Subgraph in chordal graphs.

Conclusion

Conclusion

What is known:

- SkS and DkS NP-hard in chordal graphs
- SkS and DkS approximable in chordal graphs (constant ratio)
- PTAS for DkS in interval graphs
- PTAS for SkS in proper interval graphs

Conclusion

What is known:

- SkS and DkS NP-hard in chordal graphs
- SkS and DkS approximable in chordal graphs (constant ratio)
- PTAS for DkS in interval graphs
- PTAS for SkS in proper interval graphs

What is OPEN:

- approximation schemes/lower bounds in chordal graphs ?
- complexity status in (proper) interval graphs for SkS and DkS ?
- complexity status in planar graphs for DkS ?

Merci de votre attention !

Questions ?