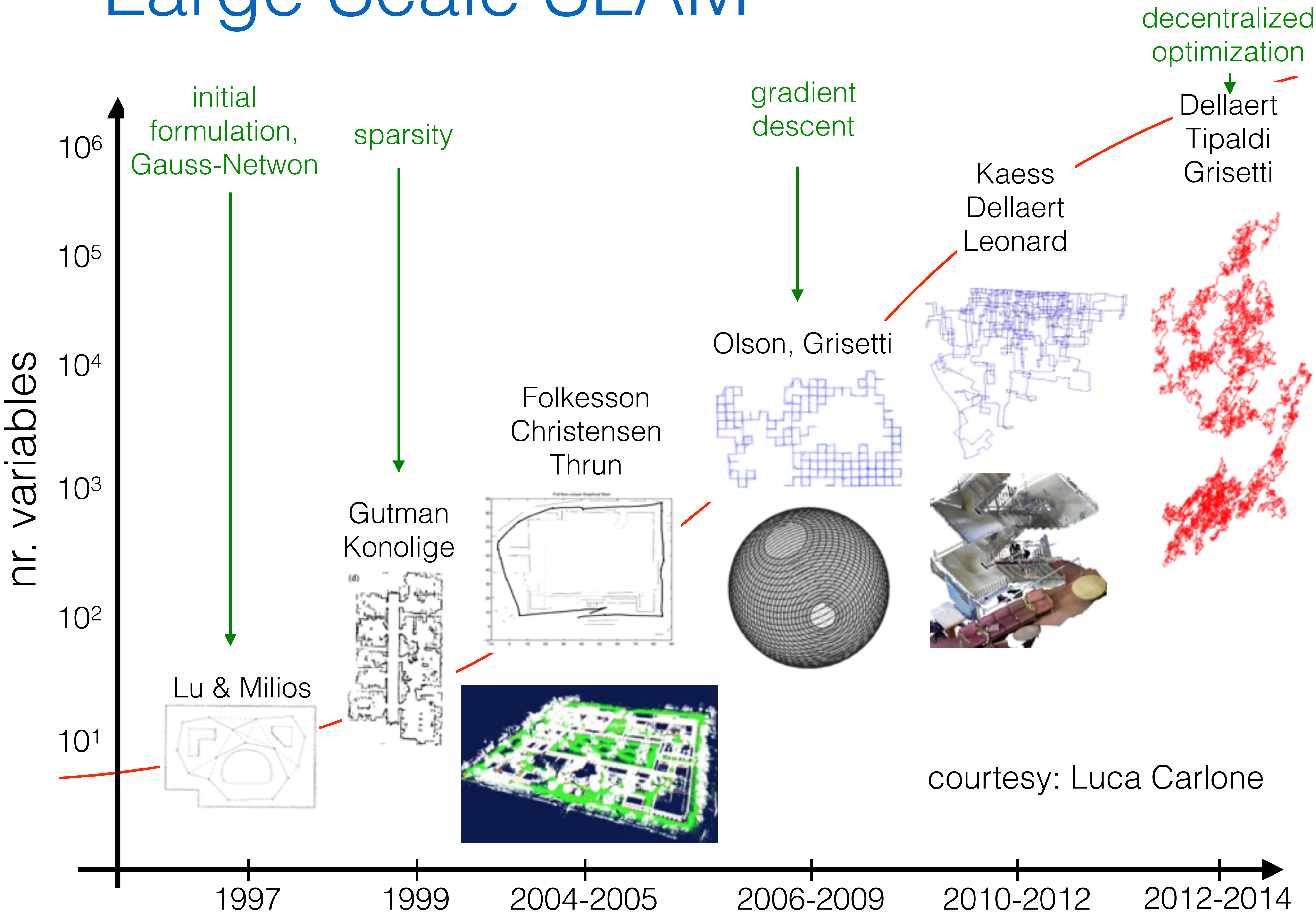
Exactly Sparse Memory Efficient SLAM using the Multi-Block Alternating Direction Method of Multipliers

Siddharth Choudhary[1], Luca Carlone[2], Henrik I. Christensen[1], Frank Dellaert[1]

[1] Institute for Robotics and Intelligent Machines, Georgia Tech
[2] Laboratory for Information and Decision Systems, MIT

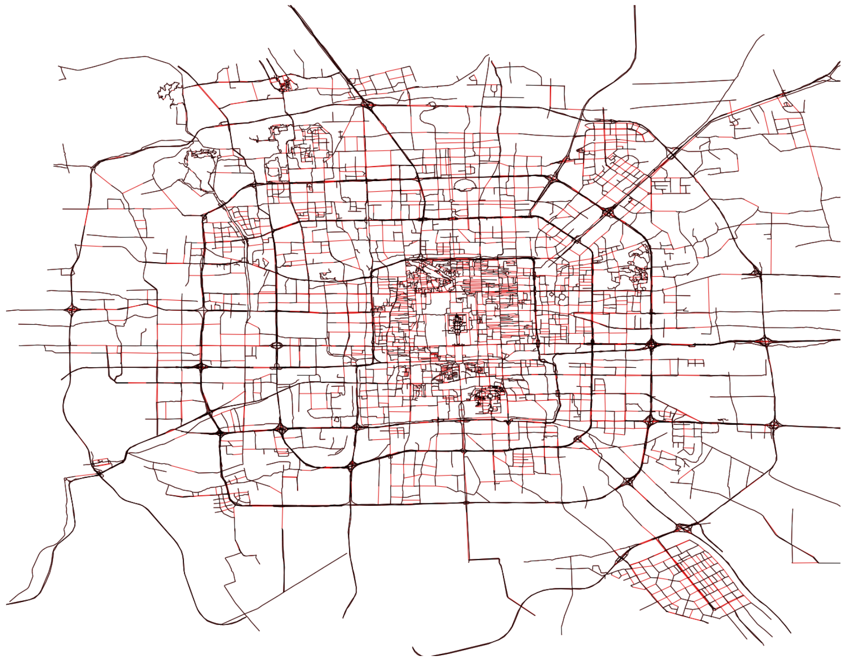
Large Scale SLAM



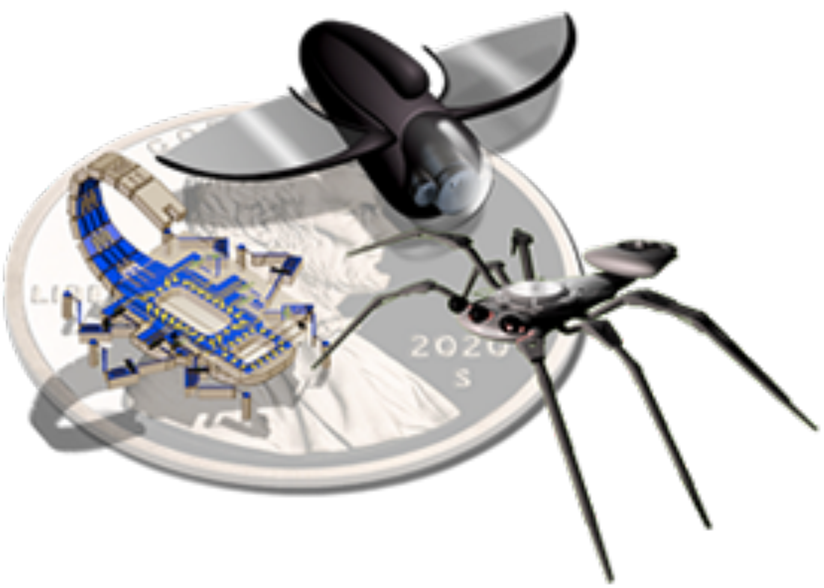
Motivation

- Existing SLAM solvers require memory linear to quadratic in the number of variables.

- Doomed to hit the memory bound:
  - city-scale/continent-scale SLAM
  - small wheeled platforms
  - small flying robots
  - low-power underwater vehicles



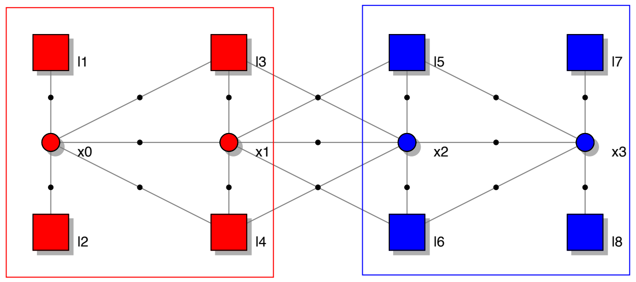
Beijing map



Related Work

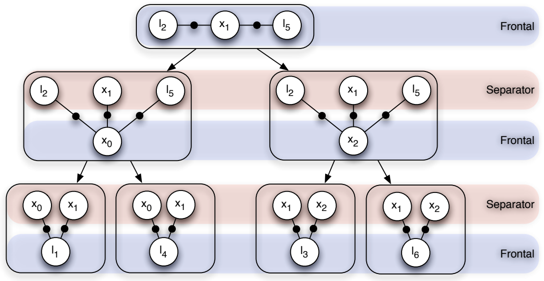
- Leonard and Feder (JOE 2001)
- Leonard and Newman (IJCAI 2003)
- Bosse et al (IJRR 2004)
- Ni et al. (ICRA 2007)

Submapping



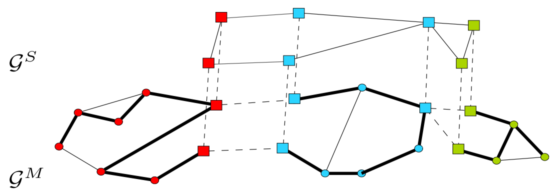
- Estrada et al. (TRO 2005)
- Frese et al. (TRO 2005)
- U. Frese (AuRo 2006)
- Ni et al. (IROS 2010)
- Grisetti et al. (ICRA 2010, IROS 2012)
- Zhao et al. (IROS 2013)

Hierarchical Optimization



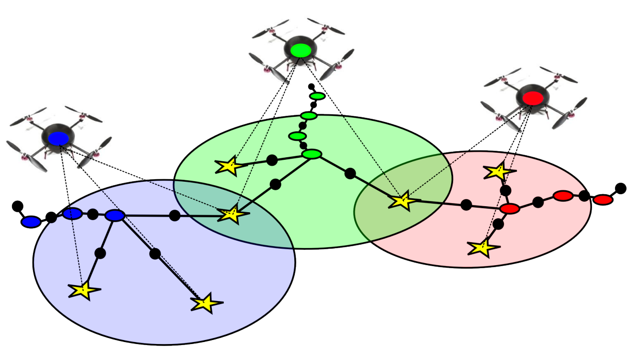
- Suger et al. (ICRA 2014)

Memory-Efficient SLAM



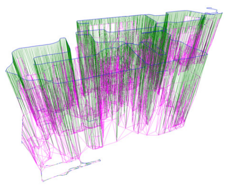
- Cunningham et al. (IROS 2010, ICRA 2013)

Distributed Optimization

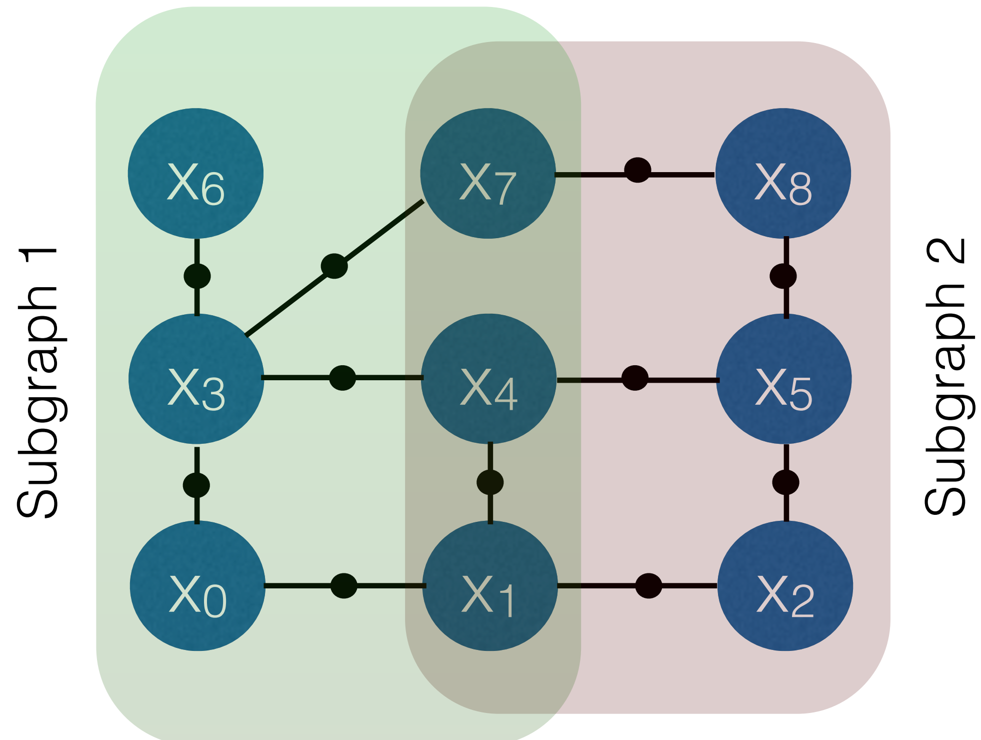


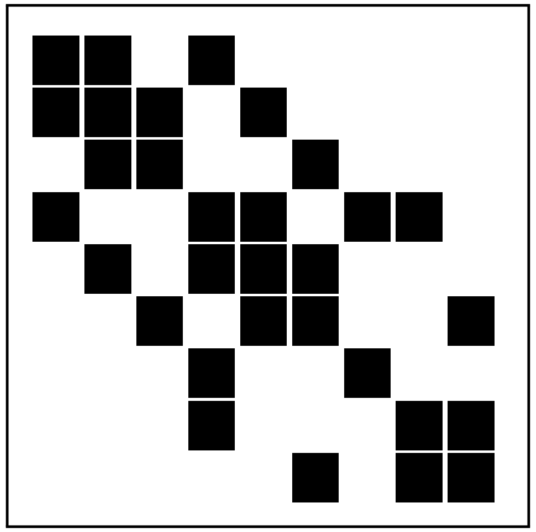
- Huang et al. (ECMR 2012)
- Wang et al. (ECMR 2013)
- Carlevaris-Bianco et al. (TRO 2014)
- Mazuran et al. (TRO 2014)

Graph Sparsification

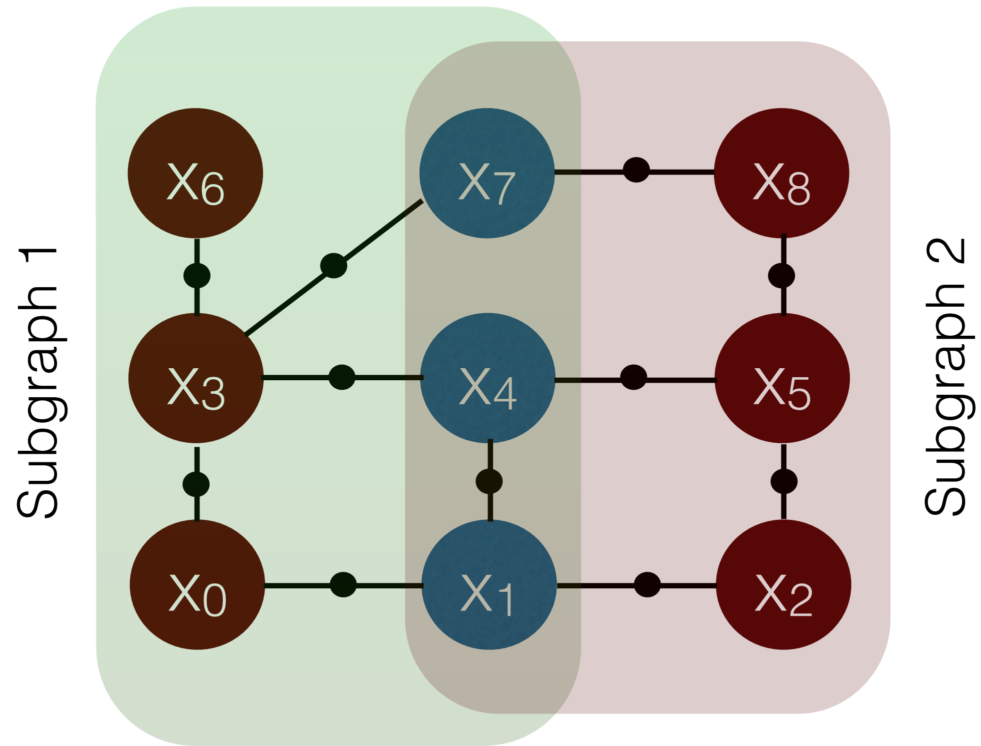


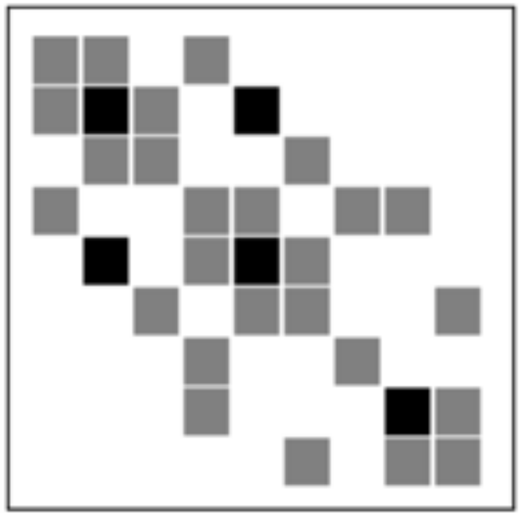
Variable Elimination



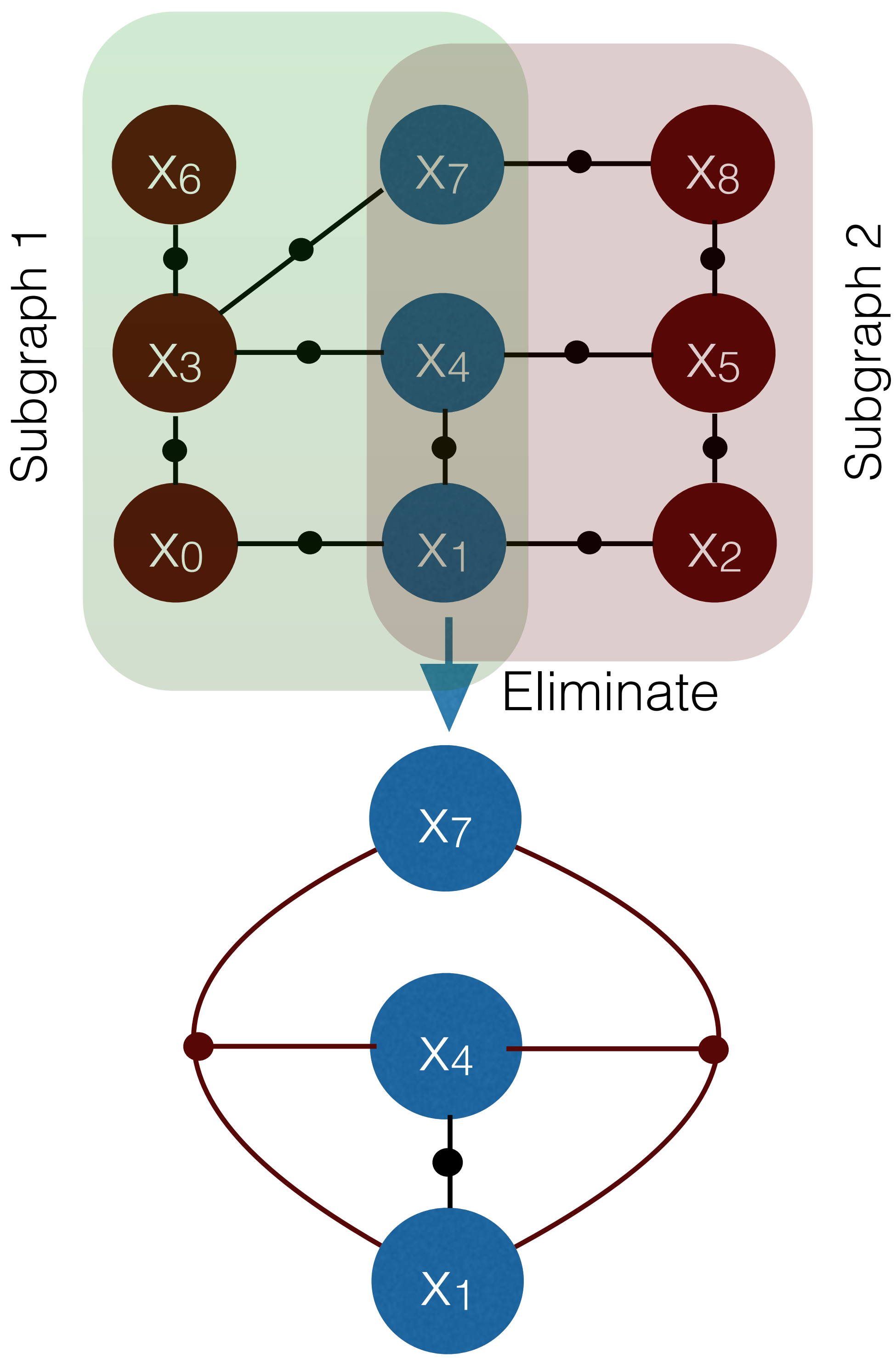


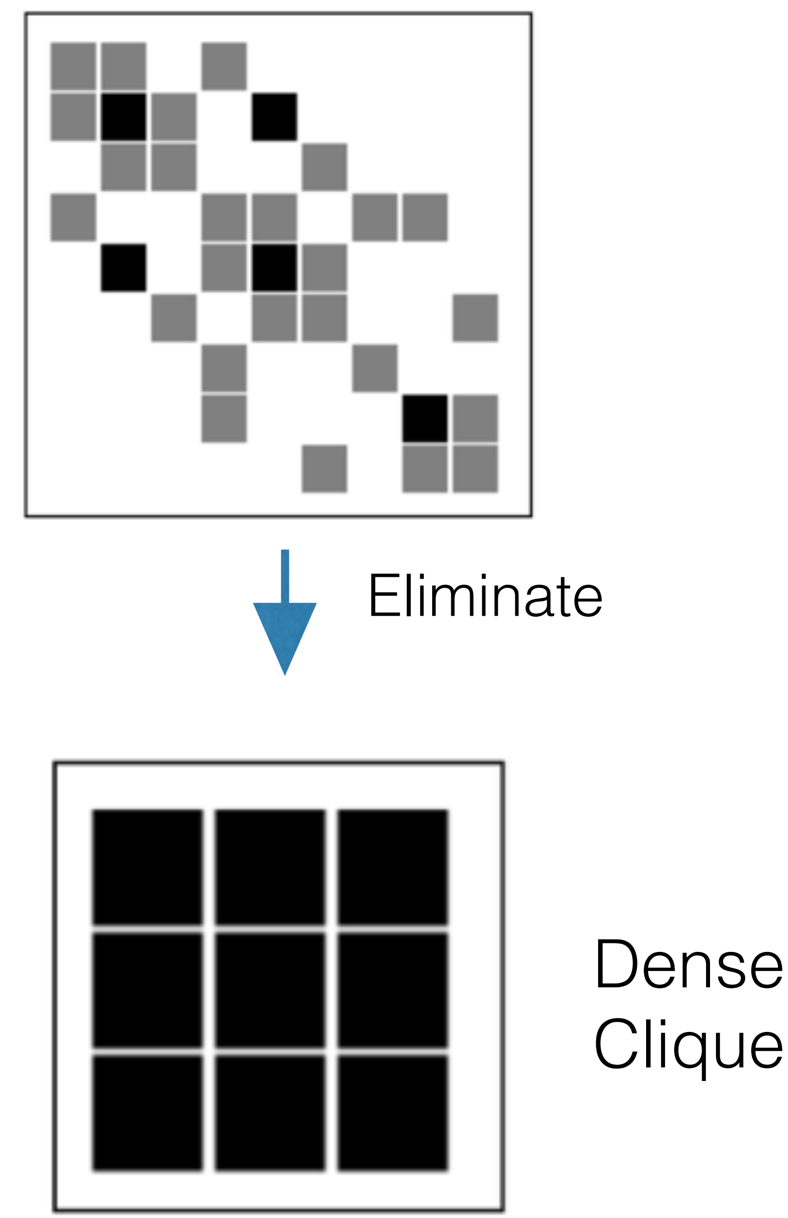
Variable Elimination





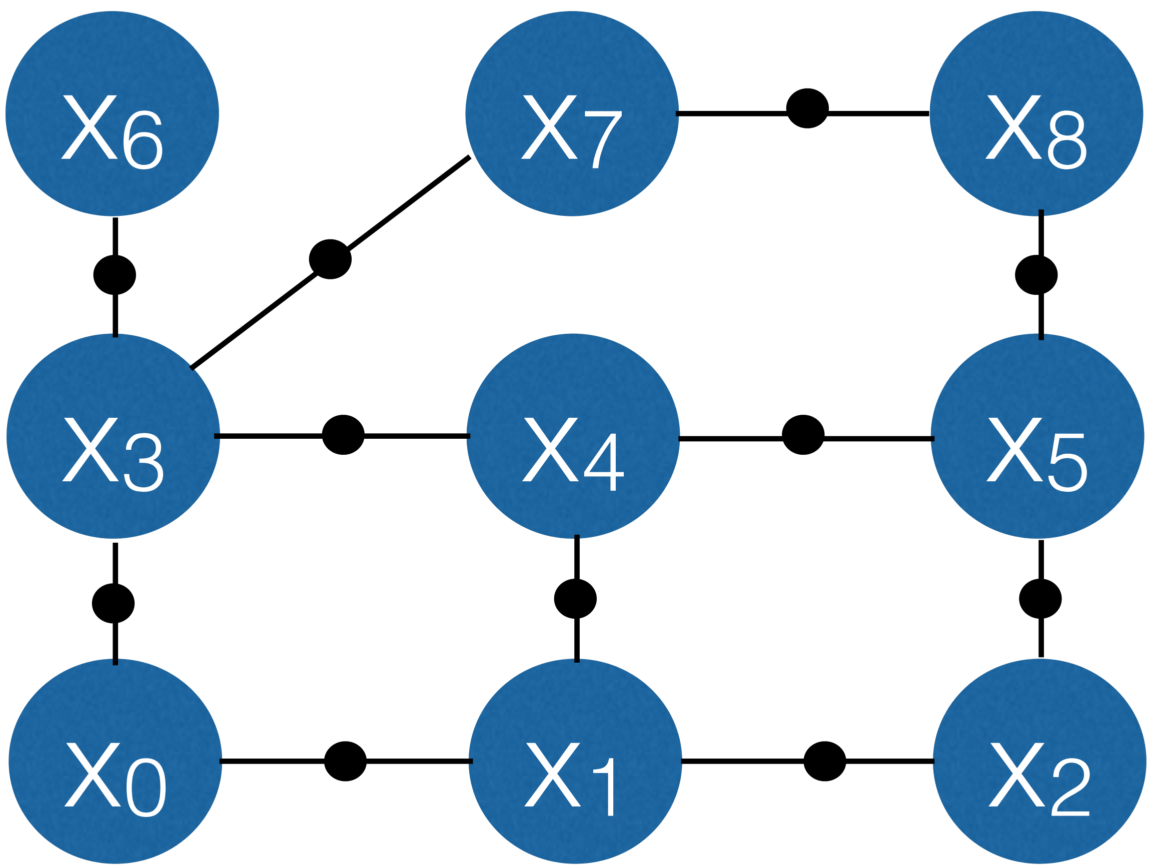
Variable Elimination



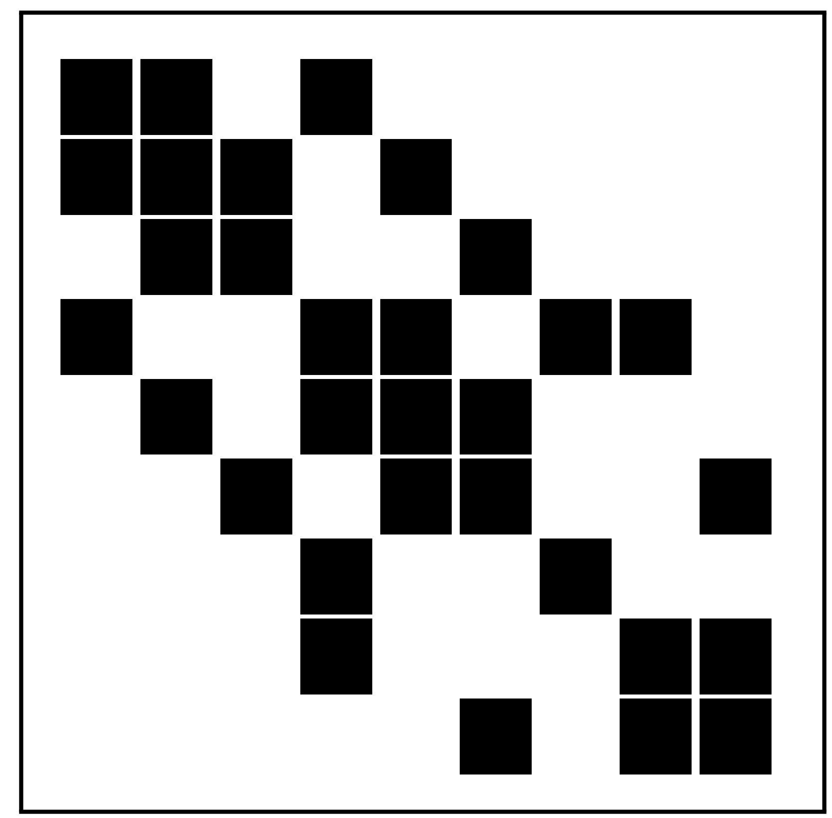


Problem Statement

Can we optimize a pose-graph within an explicit **memory bound** while preserving **sparsity**?

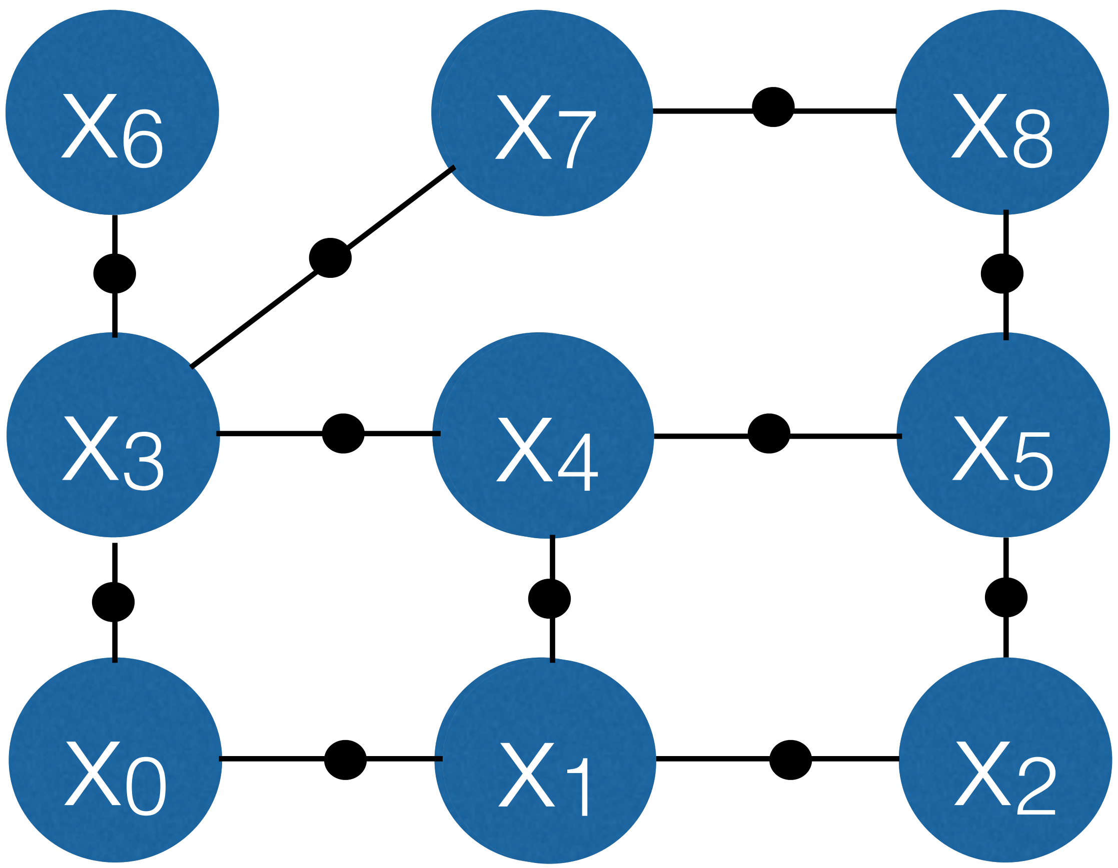


Factor Graph



Sparsity Pattern

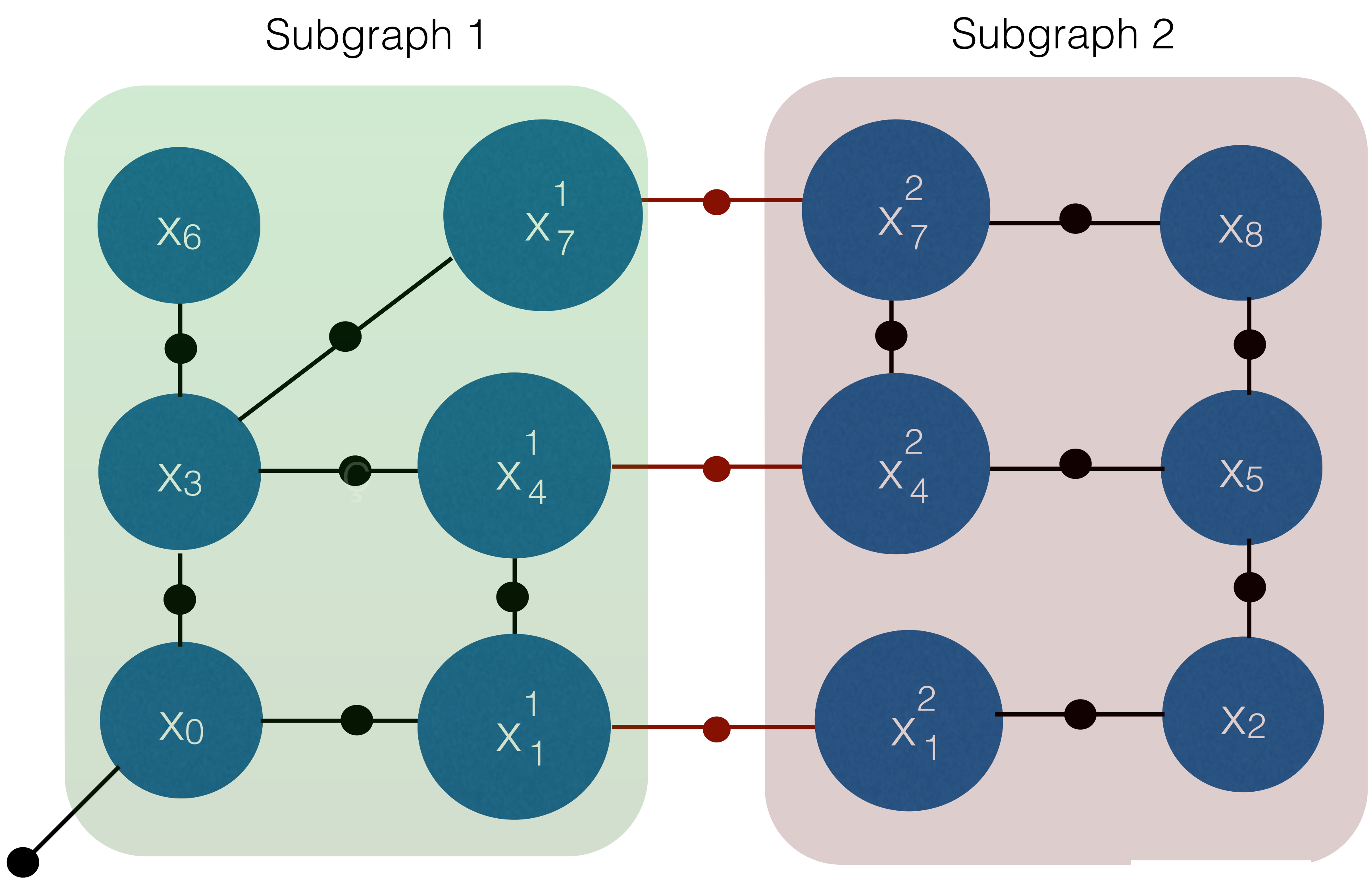
Overview of our approach



$$x^{\star} = \underset{x \in \mathrm{SE}(2)}{\arg\min}\, f(x, \mathcal{E}, \mathcal{P}) =$$

$$\underset{x \in \mathrm{SE}(2)}{\arg\min} \sum_{(i,j)\in\mathcal{E}} \left\| \mathrm{Log}\left(\bar{x}_{ij}^{-1} x_i^{-1} x_j\right) \right\|_{\Omega_{ij}}^2 + \sum_{i\in\mathcal{P}} \left\| \mathrm{Log}\left(\bar{x}_i^{-1} x_i\right) \right\|_{\Phi_{ij}}^2$$

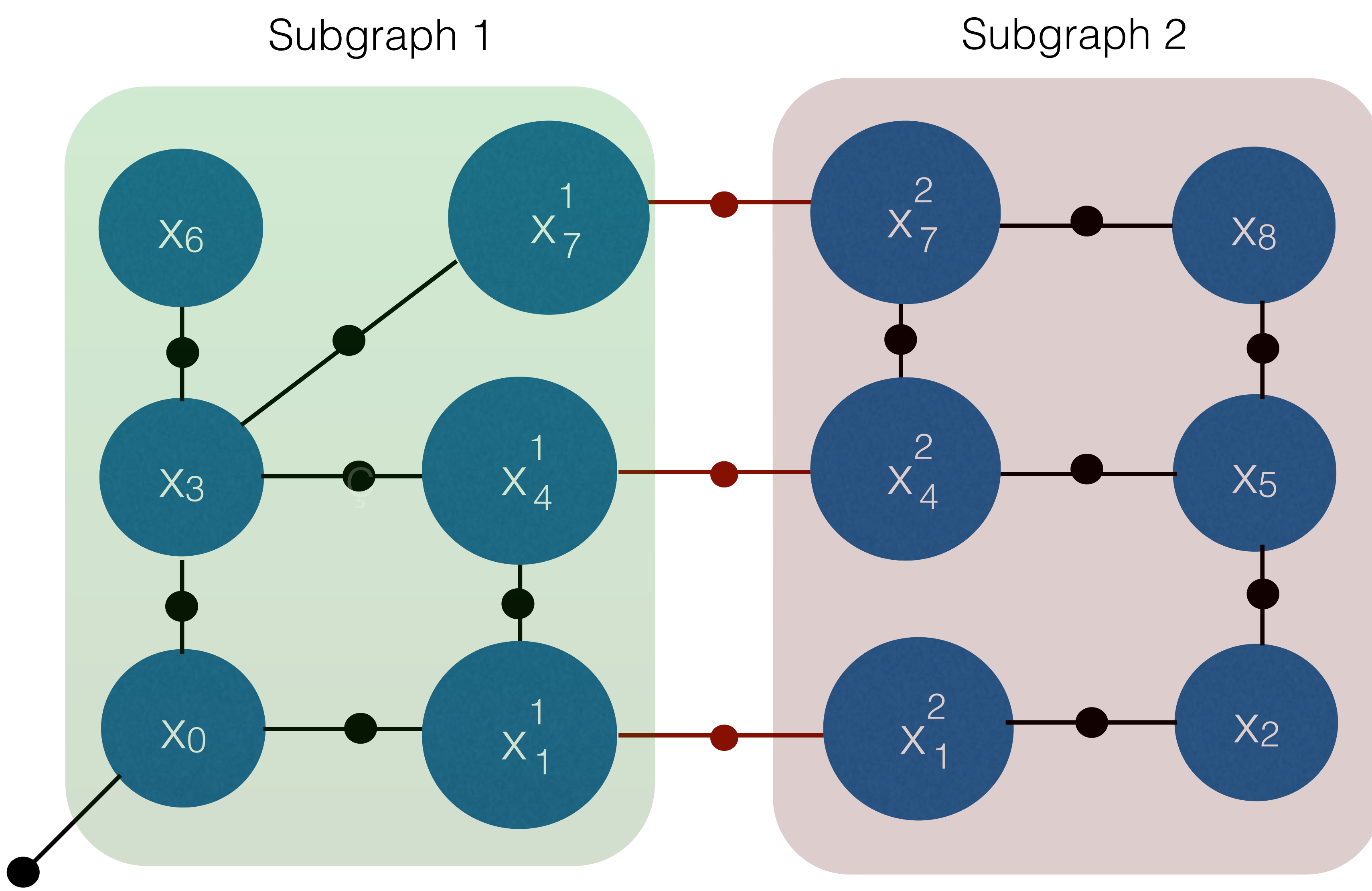
Overview of our approach



$$x^\star = \underset{x \in \mathrm{SE}(2)}{\arg\min}\, f(x, \mathcal{E}, \mathcal{P}) =$$

$$\underset{x \in \mathrm{SE}(2)}{\arg\min} \sum_{(i,j) \in \mathcal{E}} \left\| \mathrm{Log}\left(\bar{x}_{ij}^{-1} x_i^{-1} x_j \right) \right\|_{\Omega_{ij}}^2 + \sum_{i \in \mathcal{P}} \left\| \mathrm{Log}\left(\bar{x}_i^{-1} x_i \right) \right\|_{\Phi_{ij}}^2$$

Overview of our approach

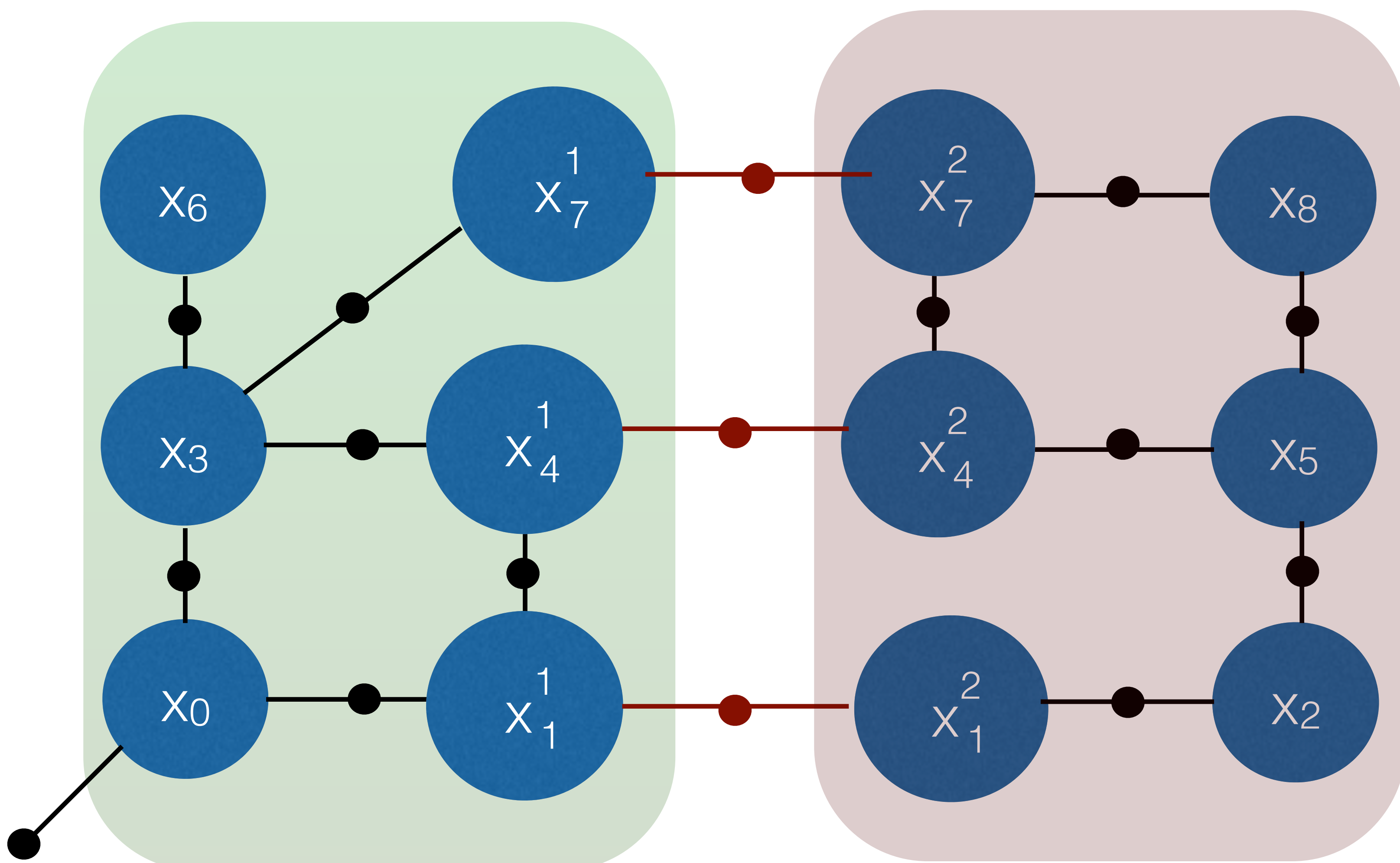


- Split separators
- Constrained Optimization
- ADMM

$$\arg\min_{x^1,\dots,x^N \in \mathrm{SE}(2)} \quad \sum_{g=1}^{N} f(x^g, \mathcal{E}^g, \mathcal{P}^g)$$

$$\text{subject to} \quad x_s^i = x_s^j, \qquad \forall s \in \mathcal{S}$$

How ADMM works?



$$\arg \min_{x^1, x^2 \in \mathrm{SE}(2)} f\left(x^1, \mathcal{E}^1, \mathcal{P}^1\right) + f\left(x^2, \mathcal{E}^2, \mathcal{P}^2\right)$$

$$\text{subject to} \quad x_s^i = x_s^j, \qquad \forall s \in \mathcal{S}$$

constrained optimization

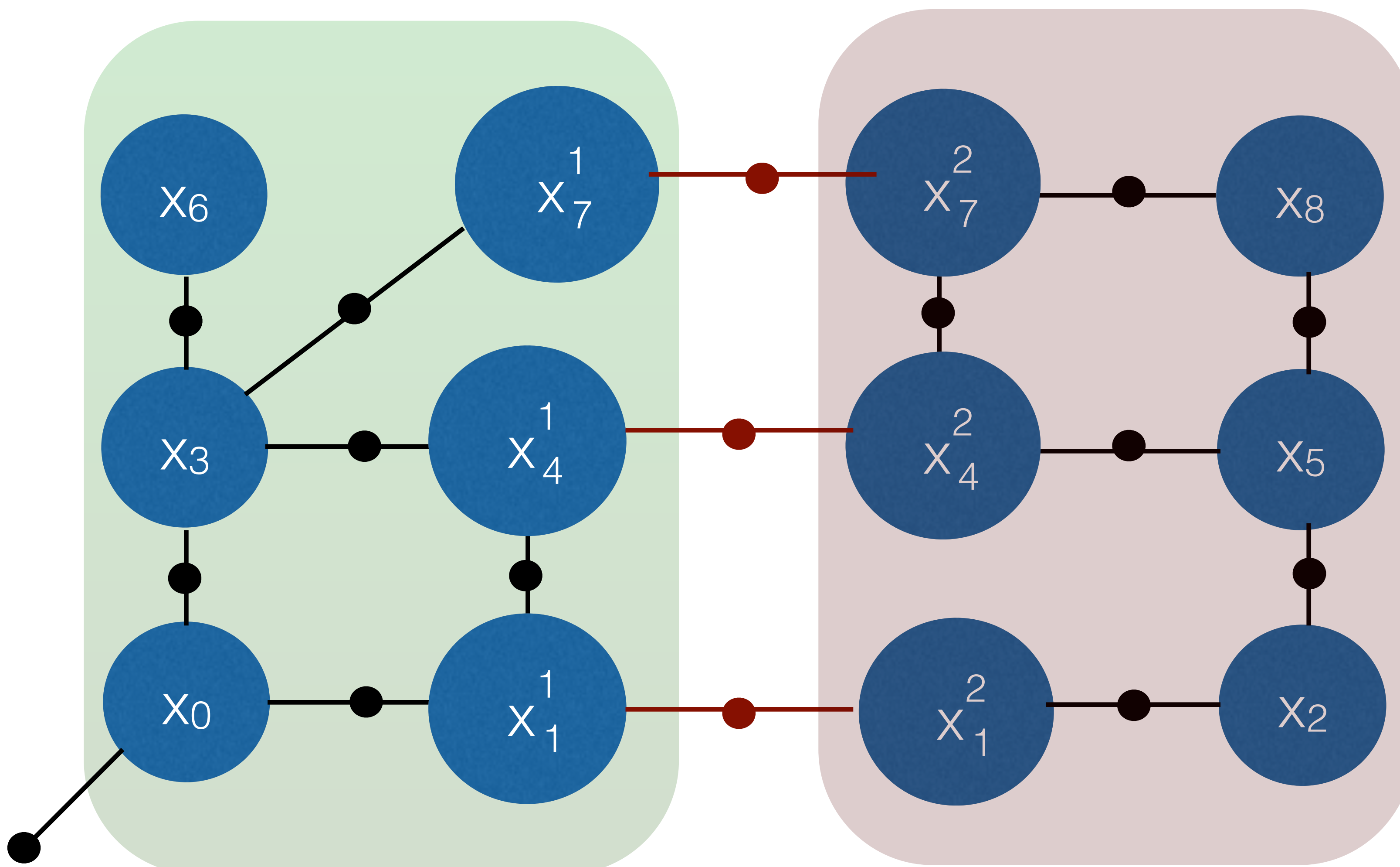
$$\arg \min_{x^1, x^2 \in \mathrm{SE}(2)} f\left(x^1, \mathcal{E}^1, \mathcal{P}^1\right) + f\left(x^2, \mathcal{E}^2, \mathcal{P}^2\right)$$

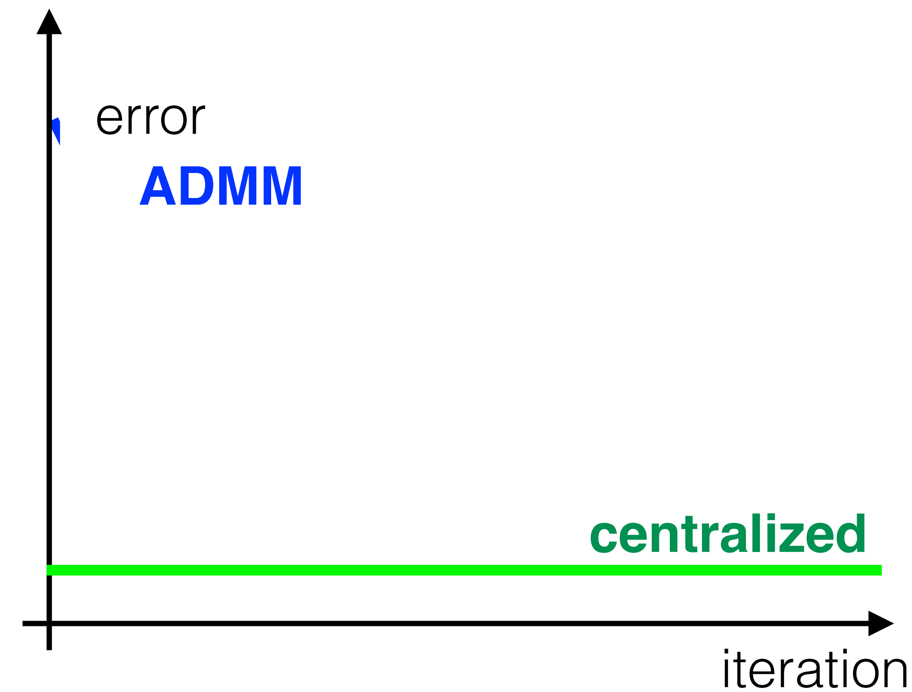
$$+ \sum_{s \in \mathcal{S}} \left[\frac{\rho}{2} \| x_s^1 - x_s^2 \|^2 + y_s^T \left(x_s^1 - x_s^2 \right) \right]$$

augmented lagrangian

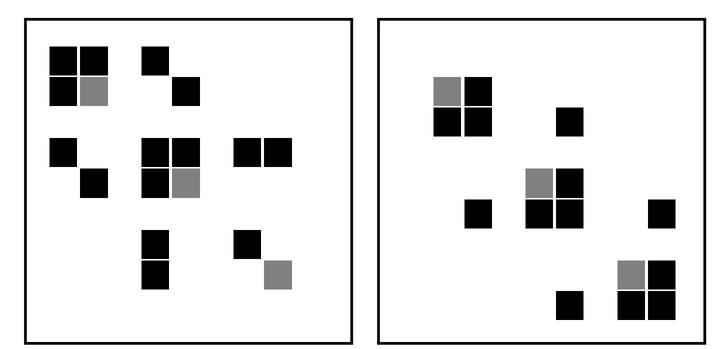
How ADMM works?

$$f\left(x^1, \mathcal{E}^1, \mathcal{P}^1\right) + f\left(x^2, \mathcal{E}^2, \mathcal{P}^2\right) + \sum_{s \in \mathcal{S}} \left[\frac{\rho}{2}\|x_s^1 - x_s^2\|^2 + y_s^T\left(x_s^1 - x_s^2\right)\right]$$



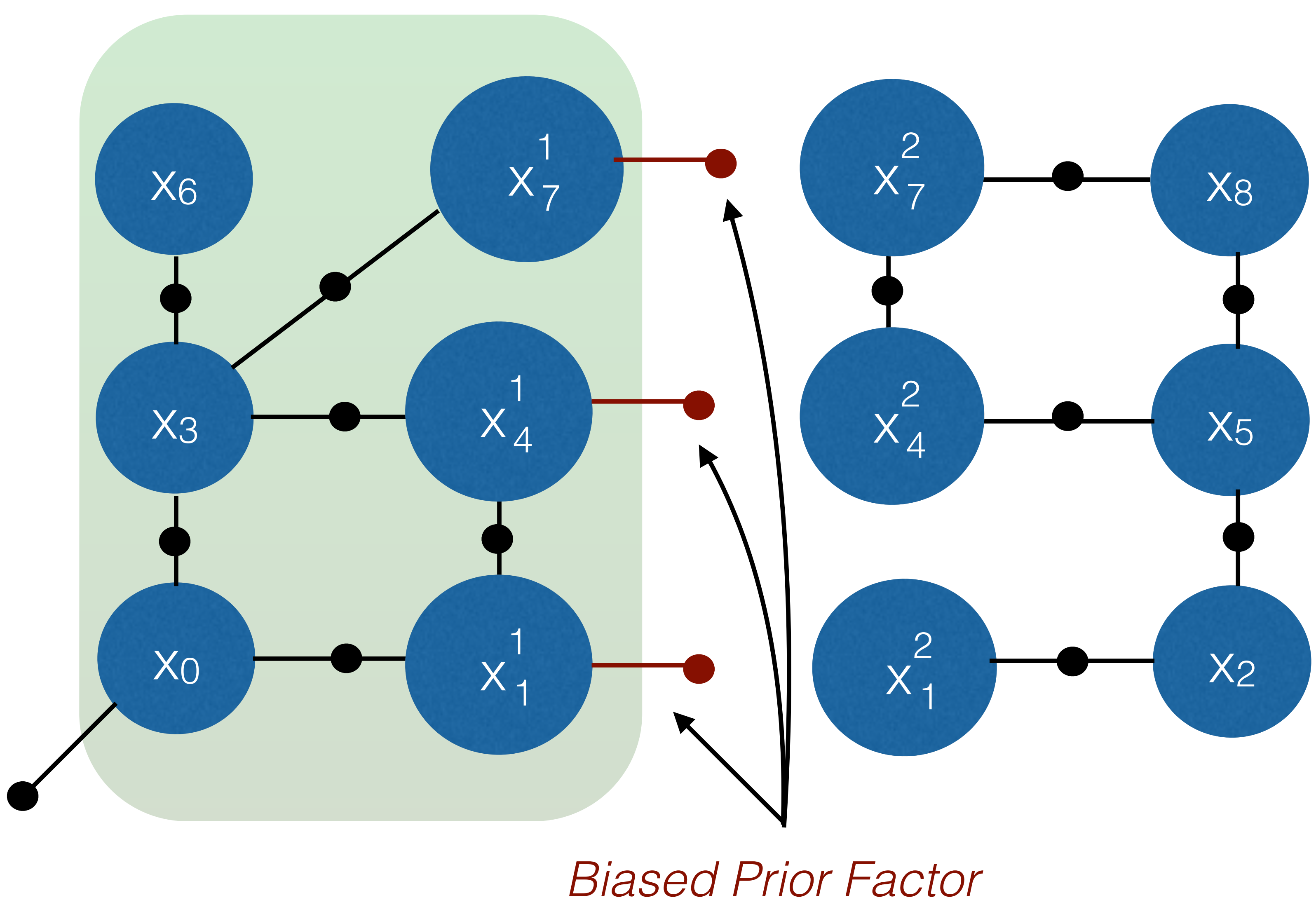


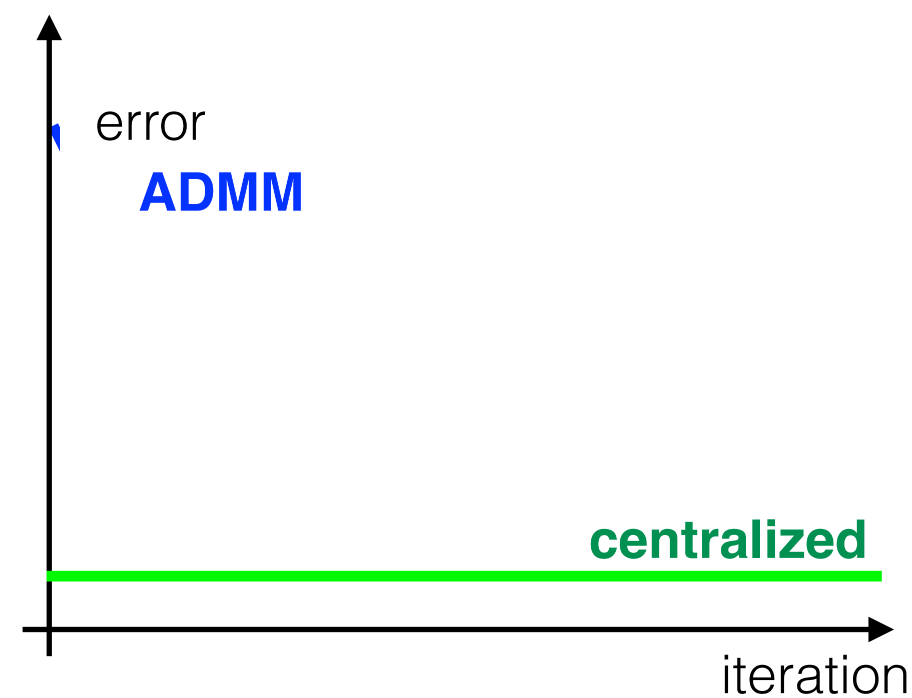
subgraph 1 subgraph 2

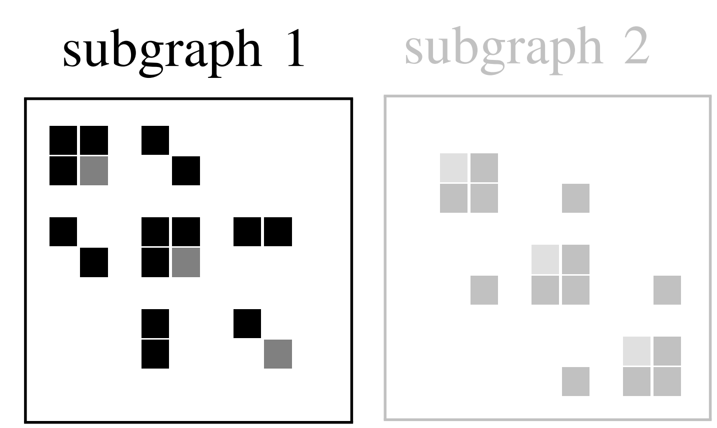


How ADMM works?

$$f\left(x^1, \mathcal{E}^1, \mathcal{P}^1\right) + f\left(x^2, \mathcal{E}^2, \mathcal{P}^2\right) + \sum_{s \in \mathcal{S}} \left[\frac{\rho}{2} \|x_s^1 - x_s^2\|^2 + y_s^T \left(x_s^1 - x_s^2\right) \right]$$

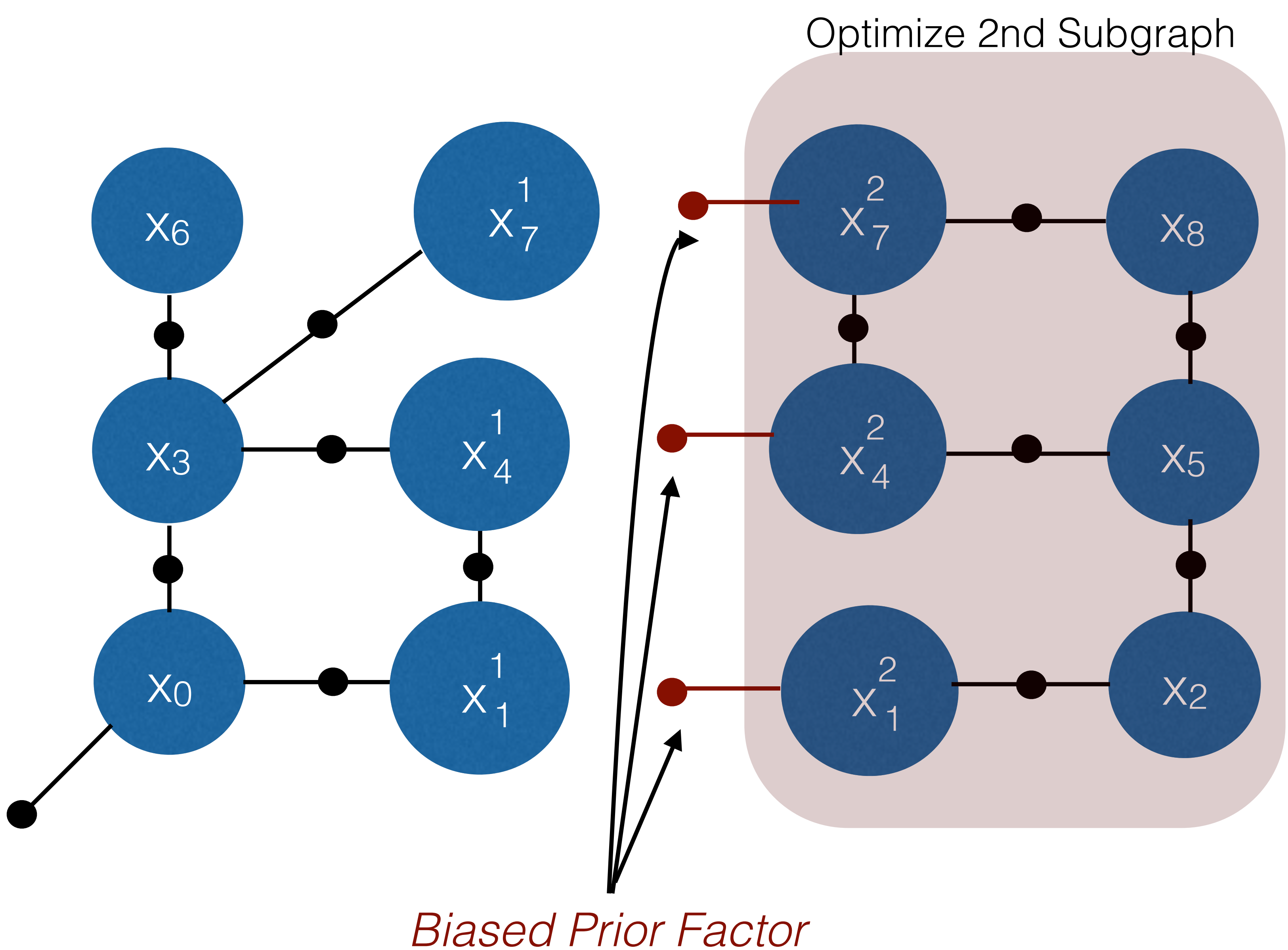


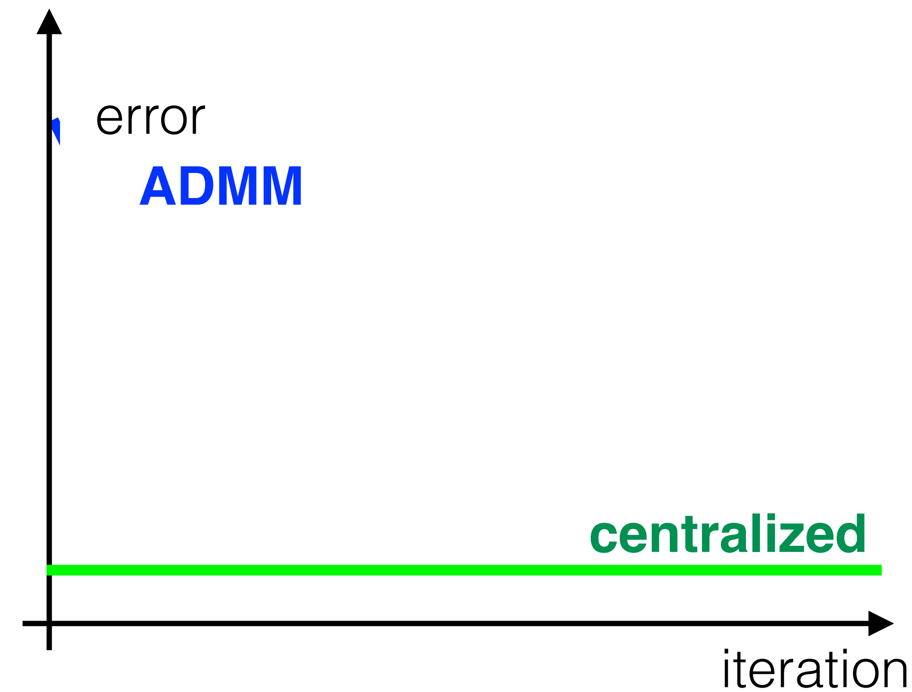


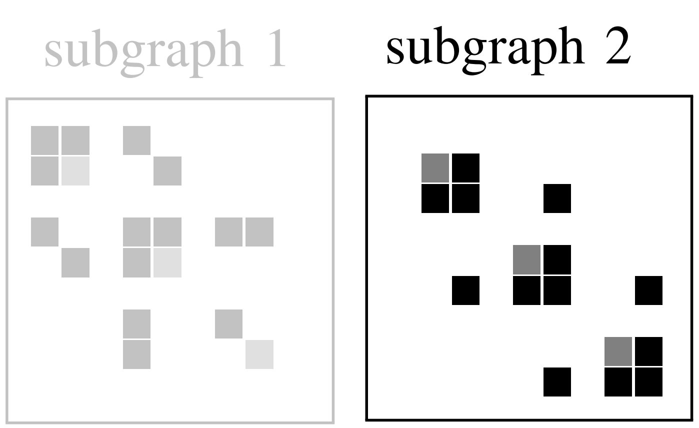


How ADMM works?

$$f\left(x^1,\mathcal{E}^1,\mathcal{P}^1\right) + \underline{f\left(x^2,\mathcal{E}^2,\mathcal{P}^2\right)} + \sum_{s\in\mathcal{S}}\left[\frac{\rho}{2}\|x_s^1 - \underline{x_s^2}\|^2 + y_s^T\left(x_s^1 - \underline{x_s^2}\right)\right]$$



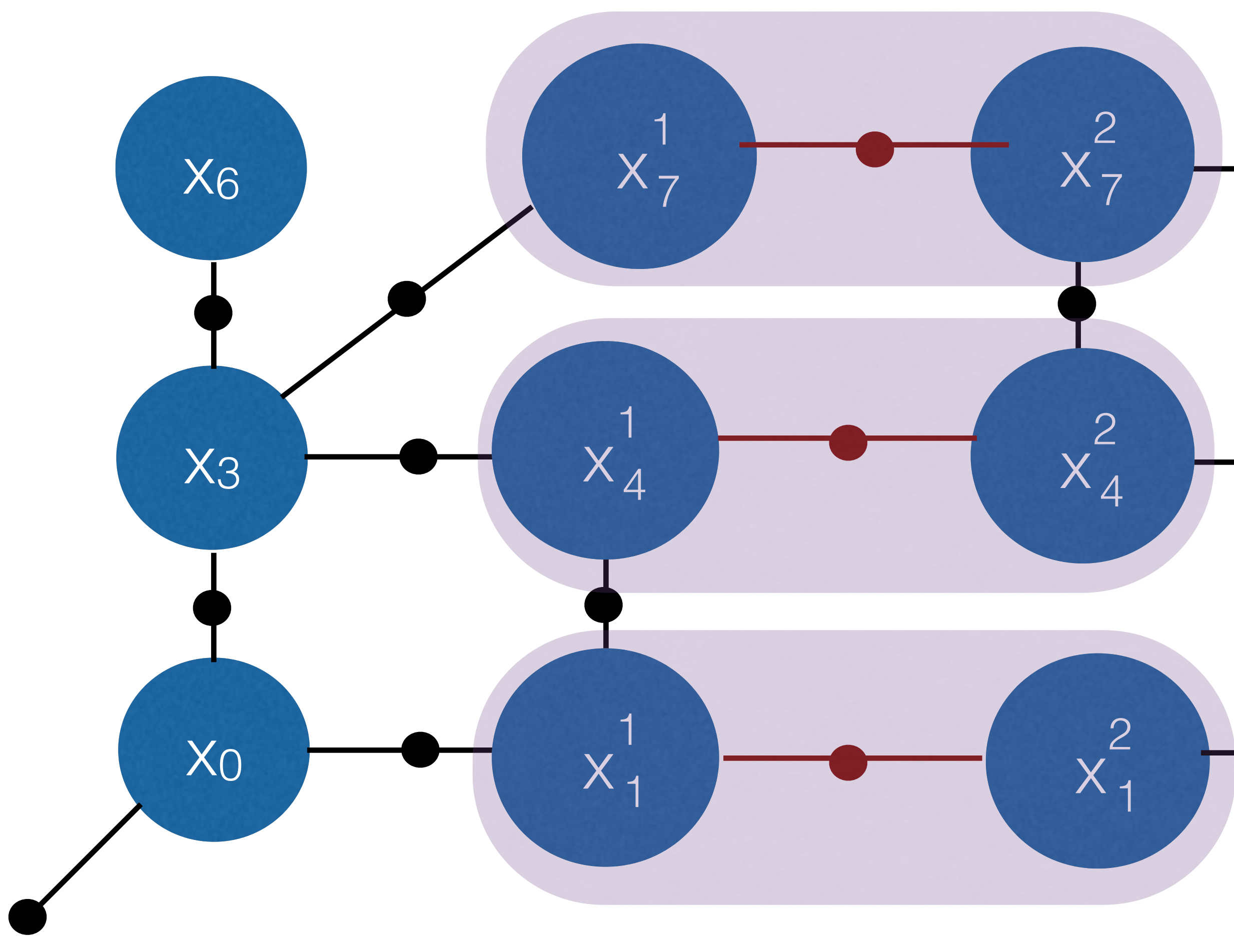


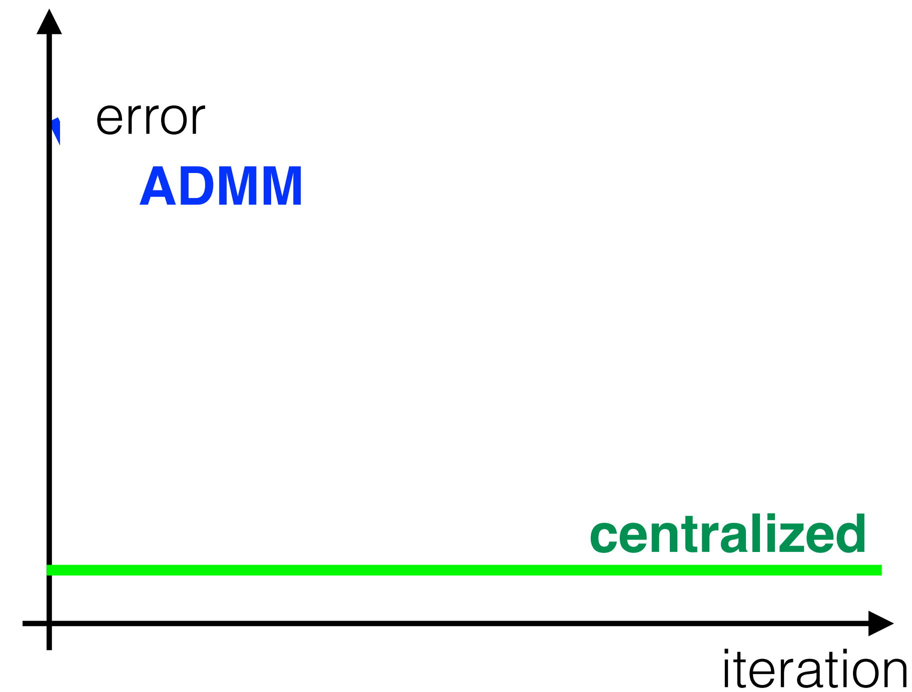


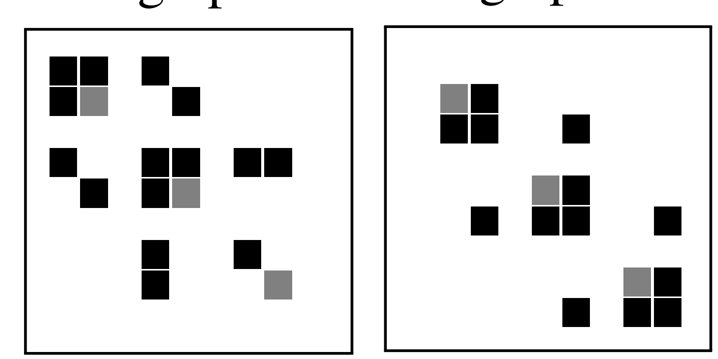
How ADMM works?

$$y_s\left(t+1\right) = y_s\left(t\right) + \sum_{s \in \mathcal{S}} \left[x_s^1 - x_s^2\right]$$

Update dual variables

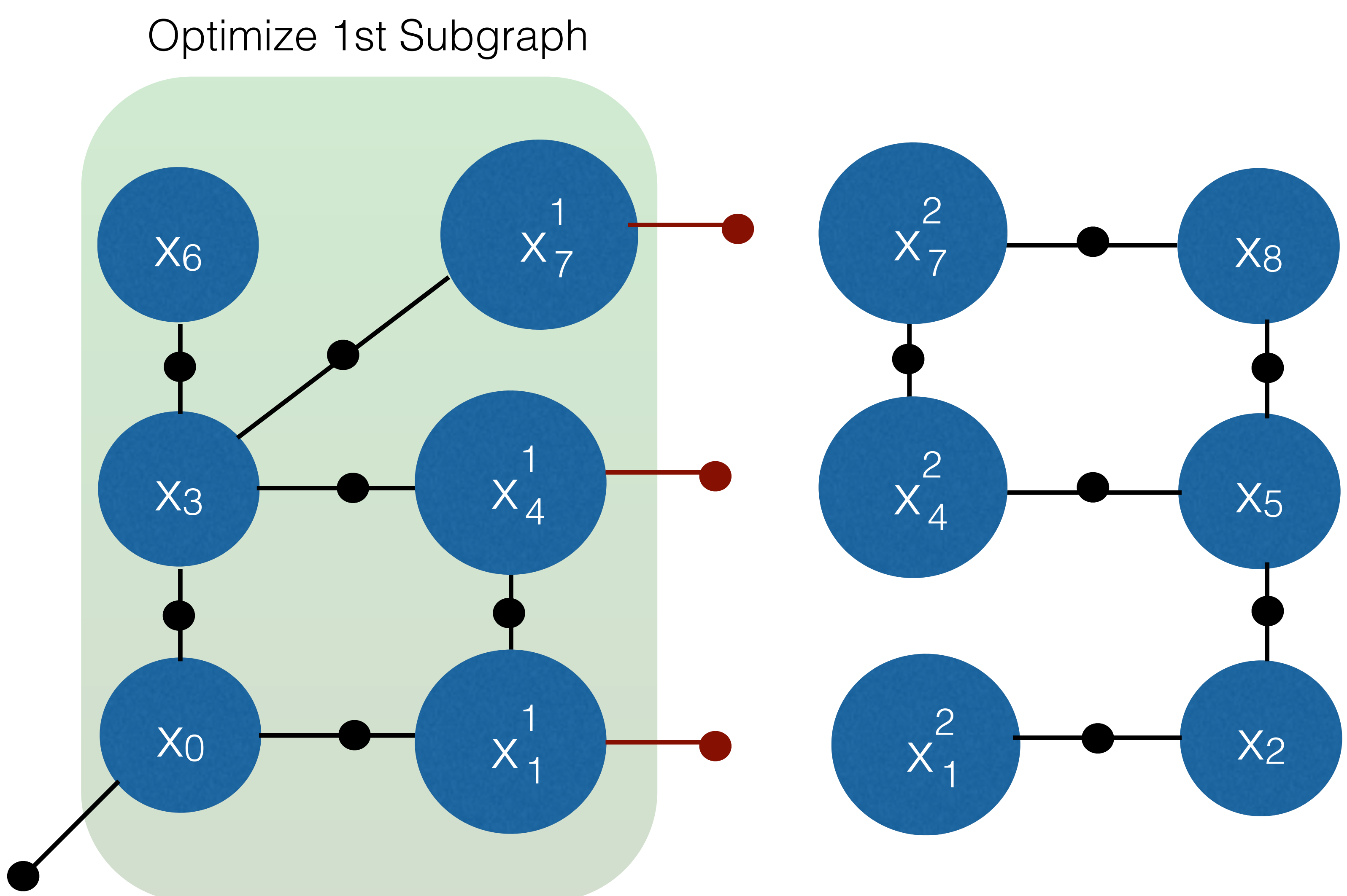


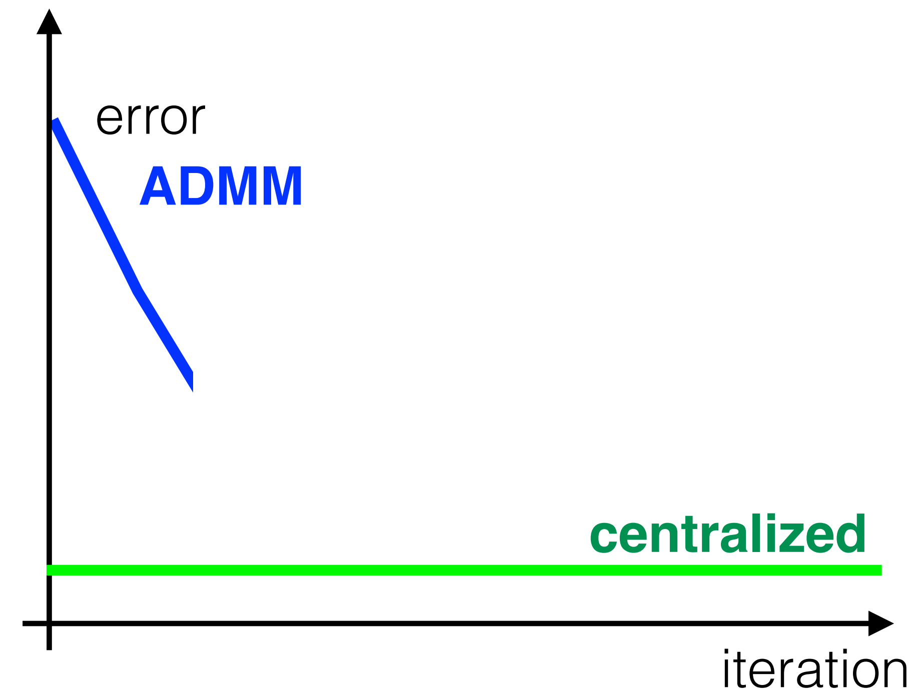


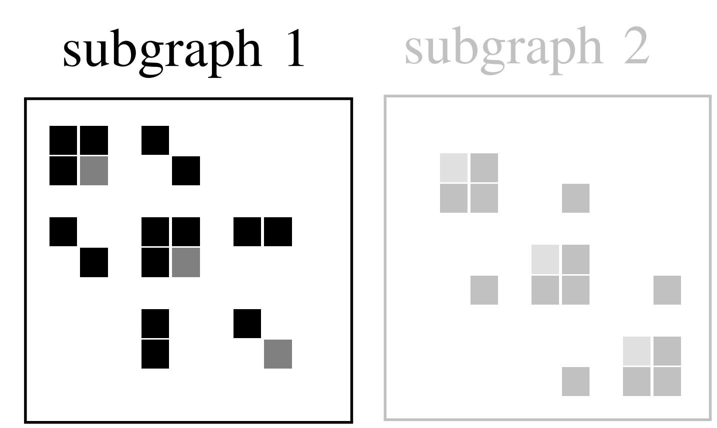


How ADMM works?

$$\underline{f\left(x^1, \mathcal{E}^1, \mathcal{P}^1\right)} + f\left(x^2, \mathcal{E}^2, \mathcal{P}^2\right) + \sum_{s \in \mathcal{S}} \left[\frac{\rho}{2} \|\underline{x_s^1} - x_s^2\|^2 + y_s^T \left(\underline{x_s^1} - x_s^2\right)\right]$$

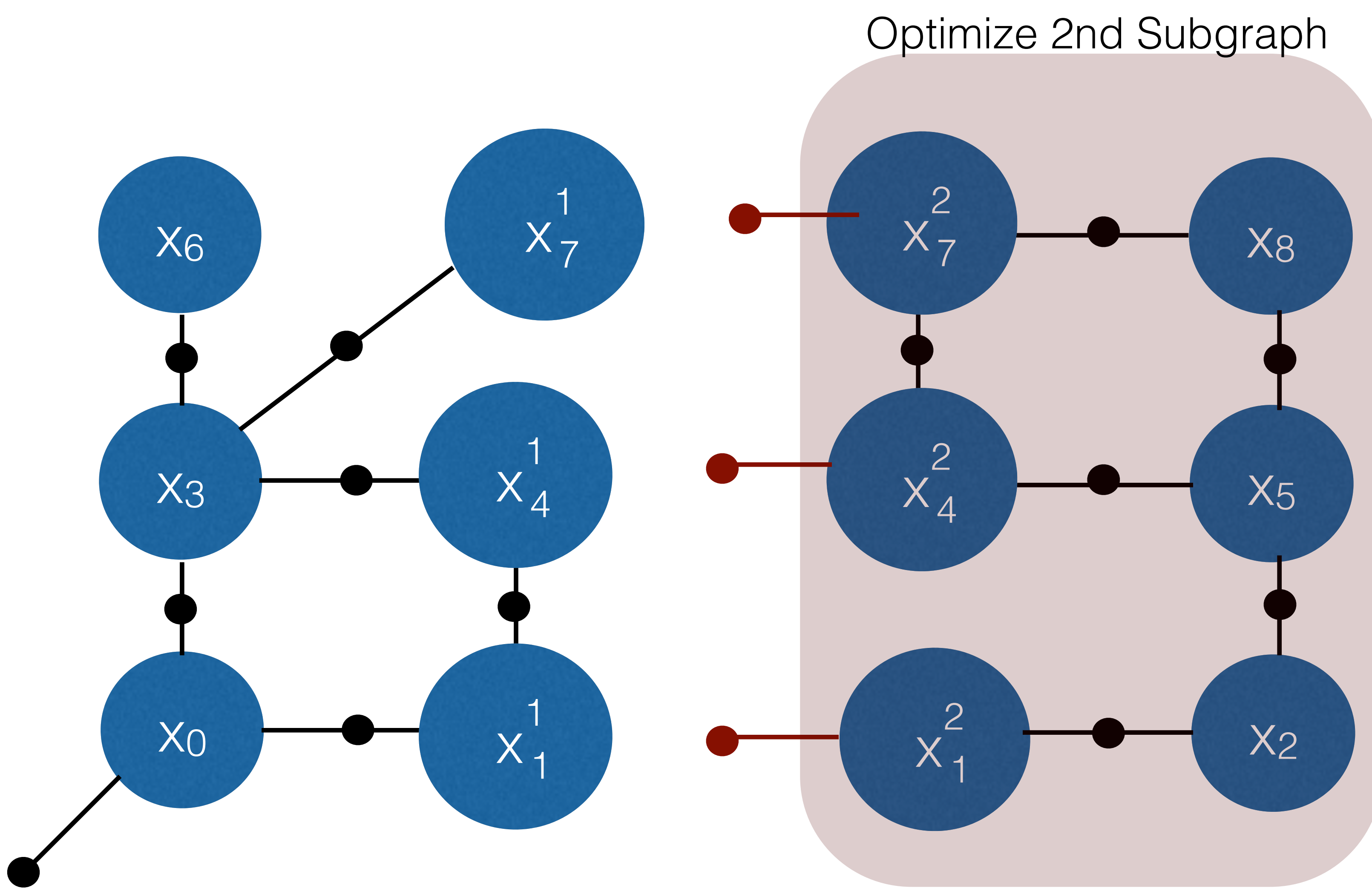


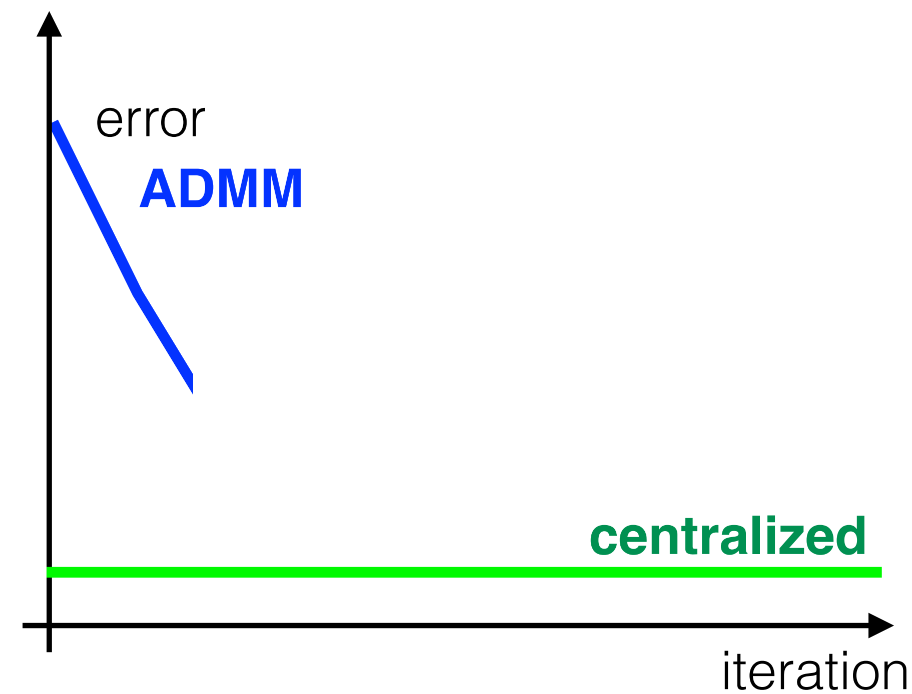


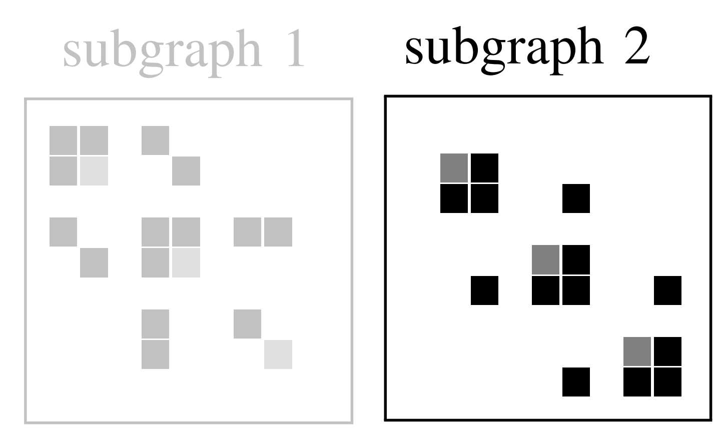


How ADMM works?

$$f\left(x^1,\mathcal{E}^1,\mathcal{P}^1\right)+\underline{f\left(x^2,\mathcal{E}^2,\mathcal{P}^2\right)}+\sum_{s\in\mathcal{S}}\left[\frac{\rho}{2}\|x_s^1-\underline{x_s^2}\|^2+y_s^T\left(x_s^1-\underline{x_s^2}\right)\right]$$

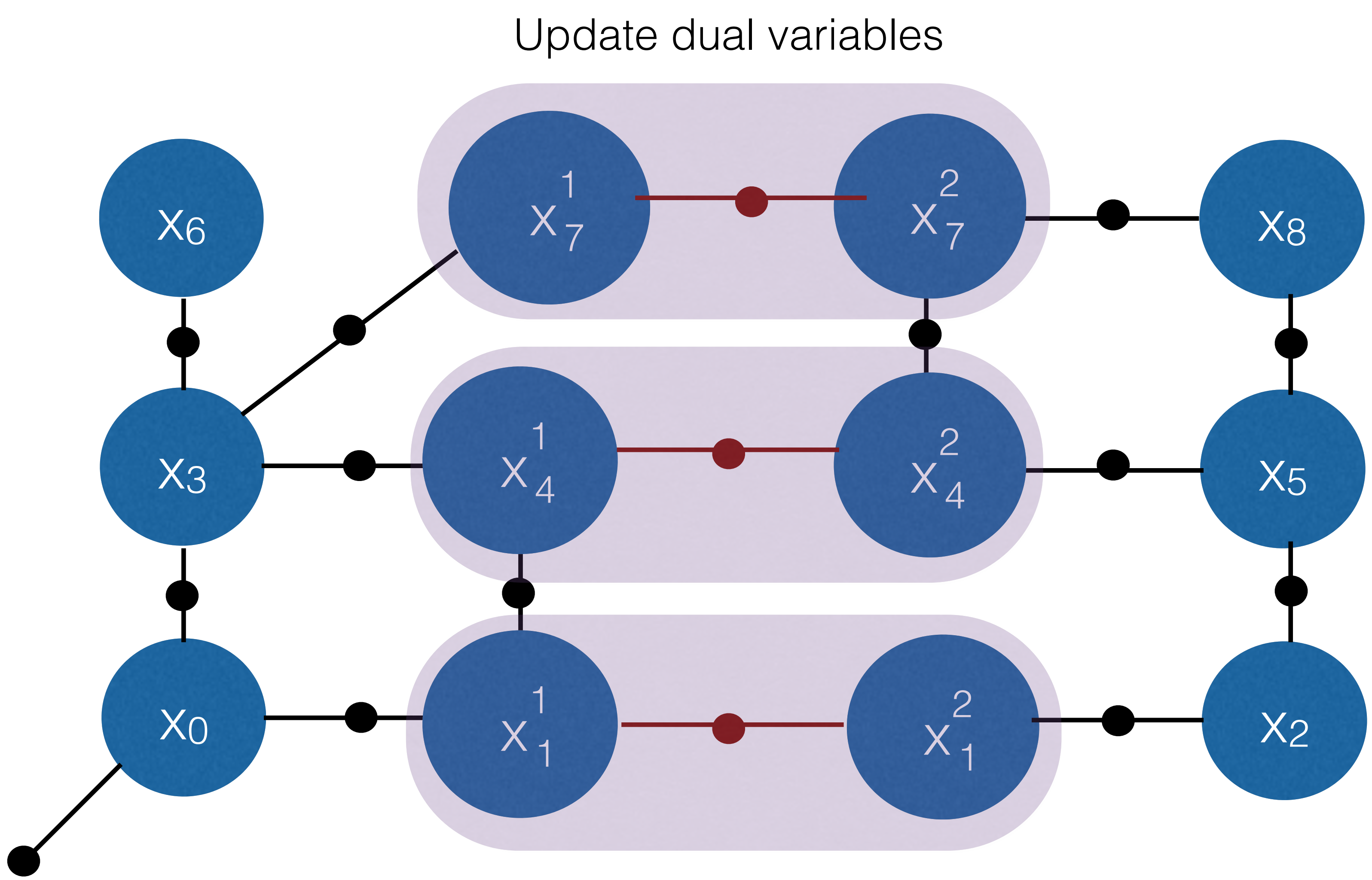


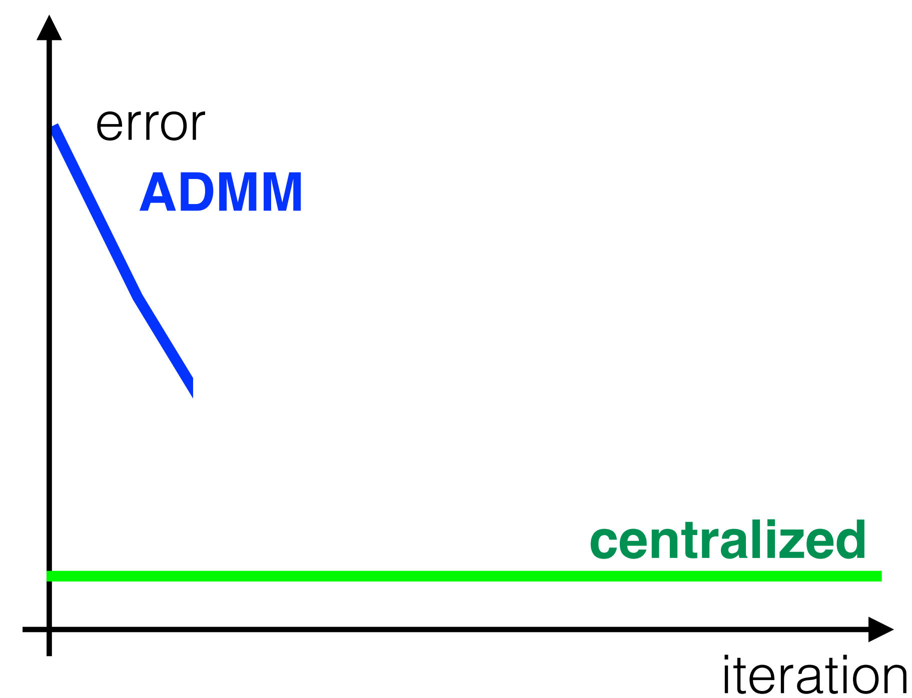


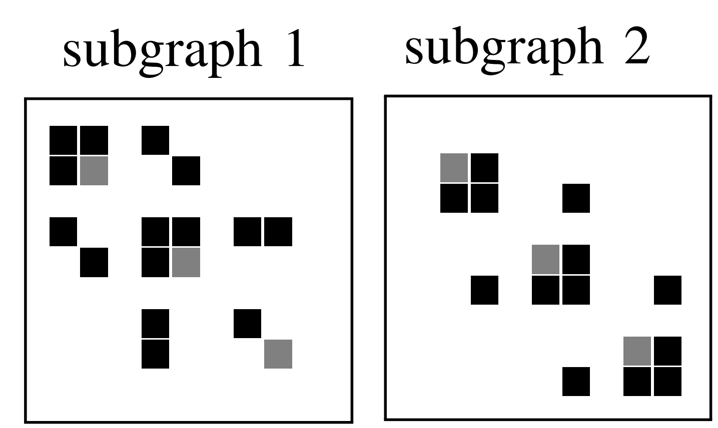


How ADMM works?

$$y_s\left(t+1\right) = y_s\left(t\right) + \underline{\sum_{s\in\mathcal{S}}\left[x_s^1 - x_s^2\right]}$$



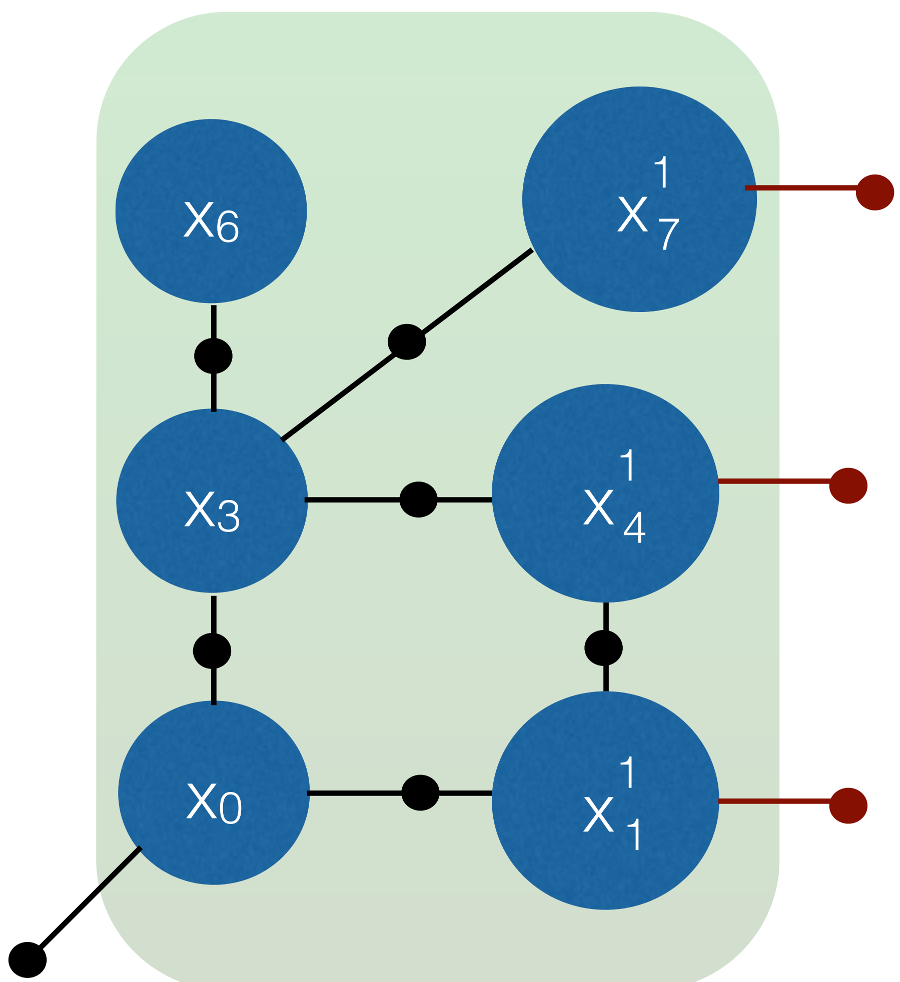


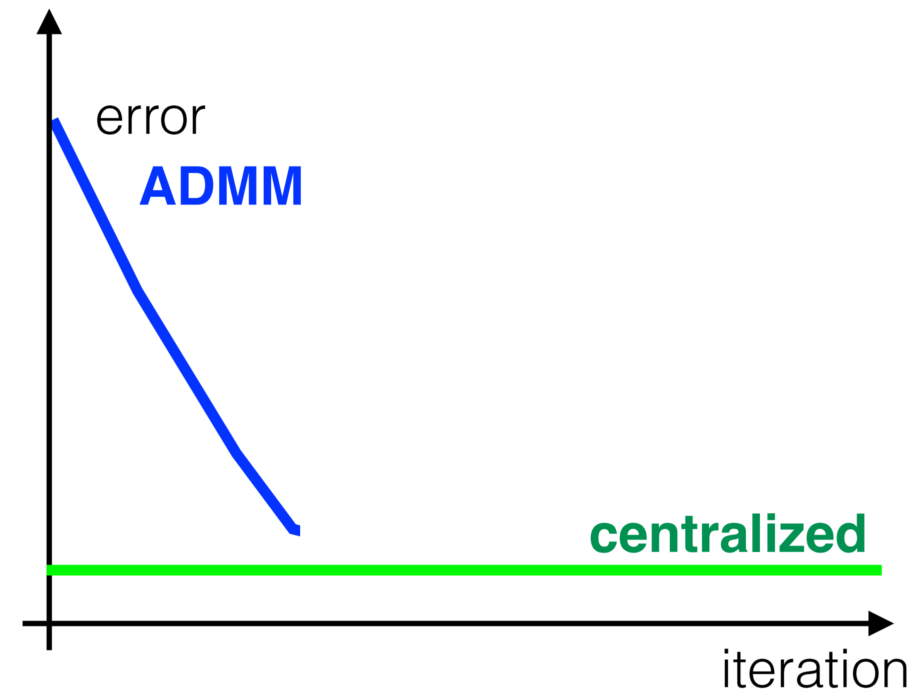


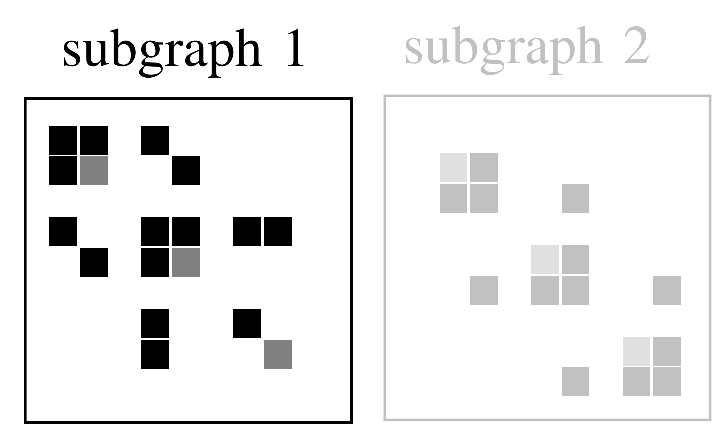
How ADMM works?

$$\underline{f\left(x^1, \mathcal{E}^1, \mathcal{P}^1\right)} + f\left(x^2, \mathcal{E}^2, \mathcal{P}^2\right) + \sum_{s \in \mathcal{S}} \left[\frac{\rho}{2}\|\underline{x_s^1} - x_s^2\|^2 + y_s^T\left(\underline{x_s^1} - x_s^2\right)\right]$$

Optimize 1st Subgraph

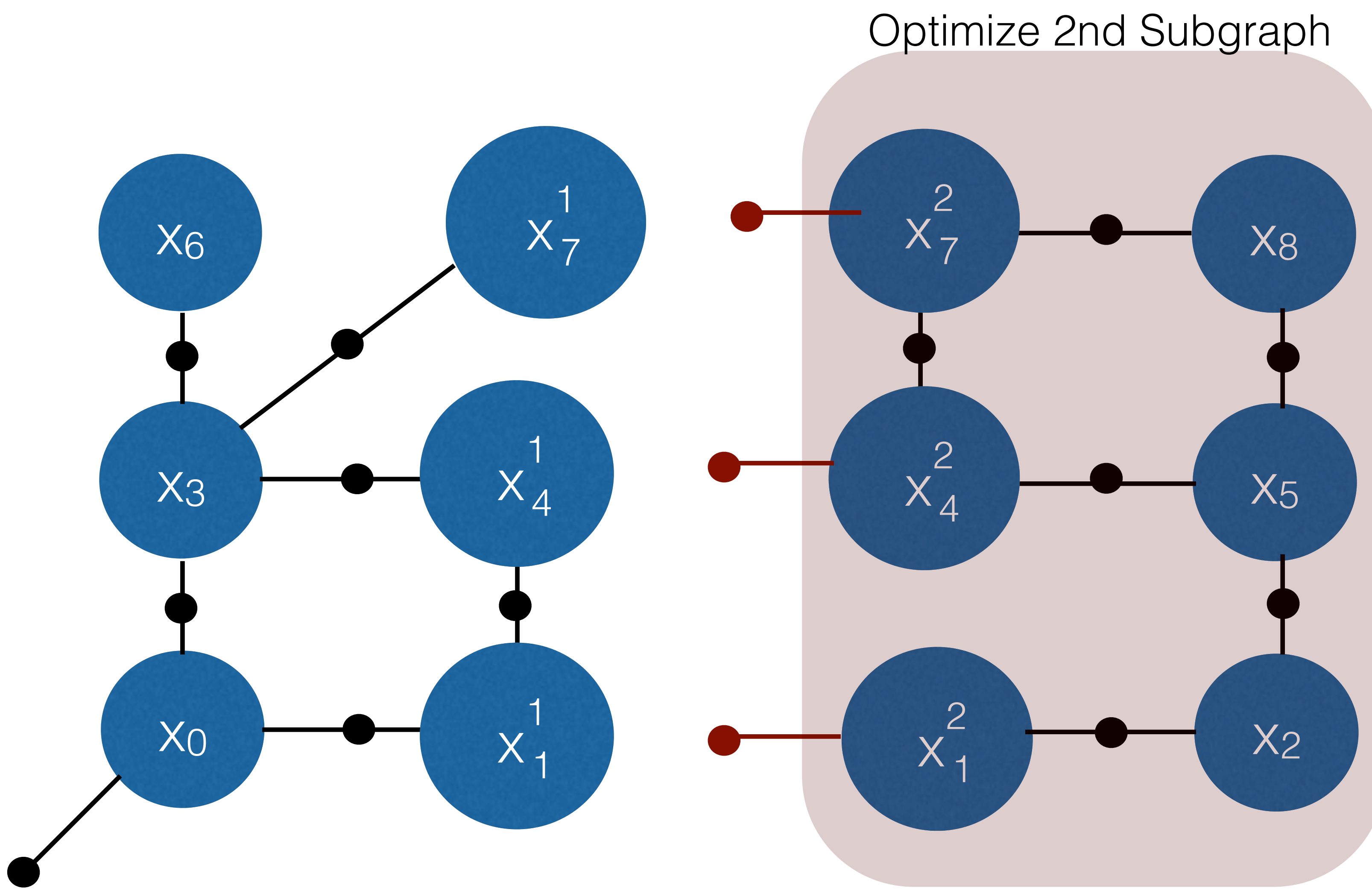


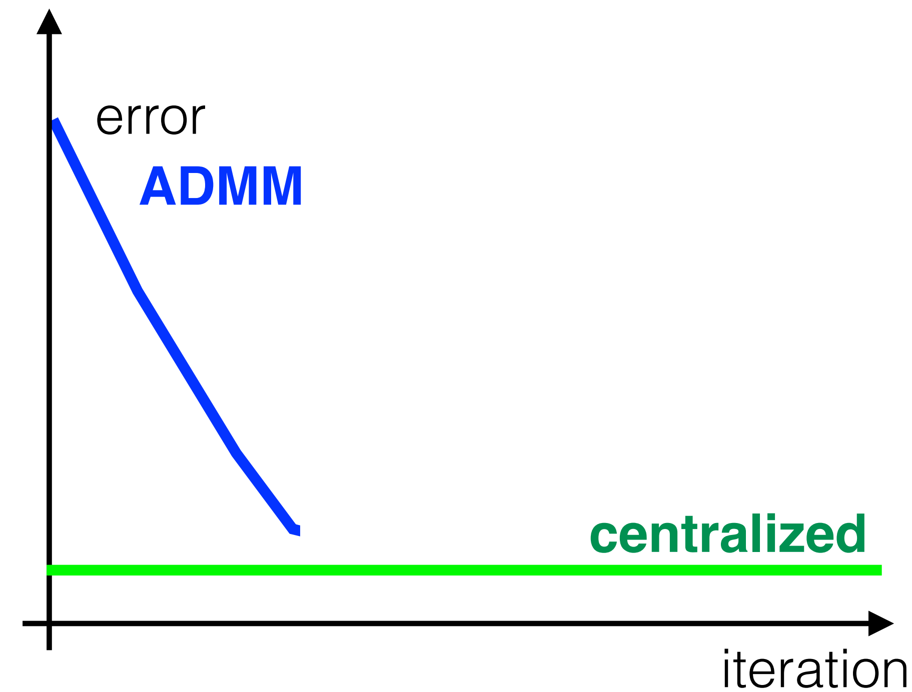


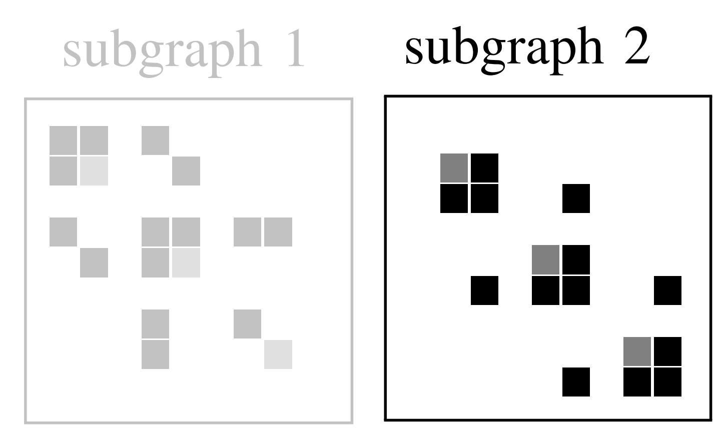


How ADMM works?

$$f\left(x^1,\mathcal{E}^1,\mathcal{P}^1\right)+\underline{f\left(x^2,\mathcal{E}^2,\mathcal{P}^2\right)}+\sum_{s\in\mathcal{S}}\left[\frac{\rho}{2}\|x_s^1-\underline{x_s^2}\|^2+y_s^T\left(x_s^1-\underline{x_s^2}\right)\right]$$



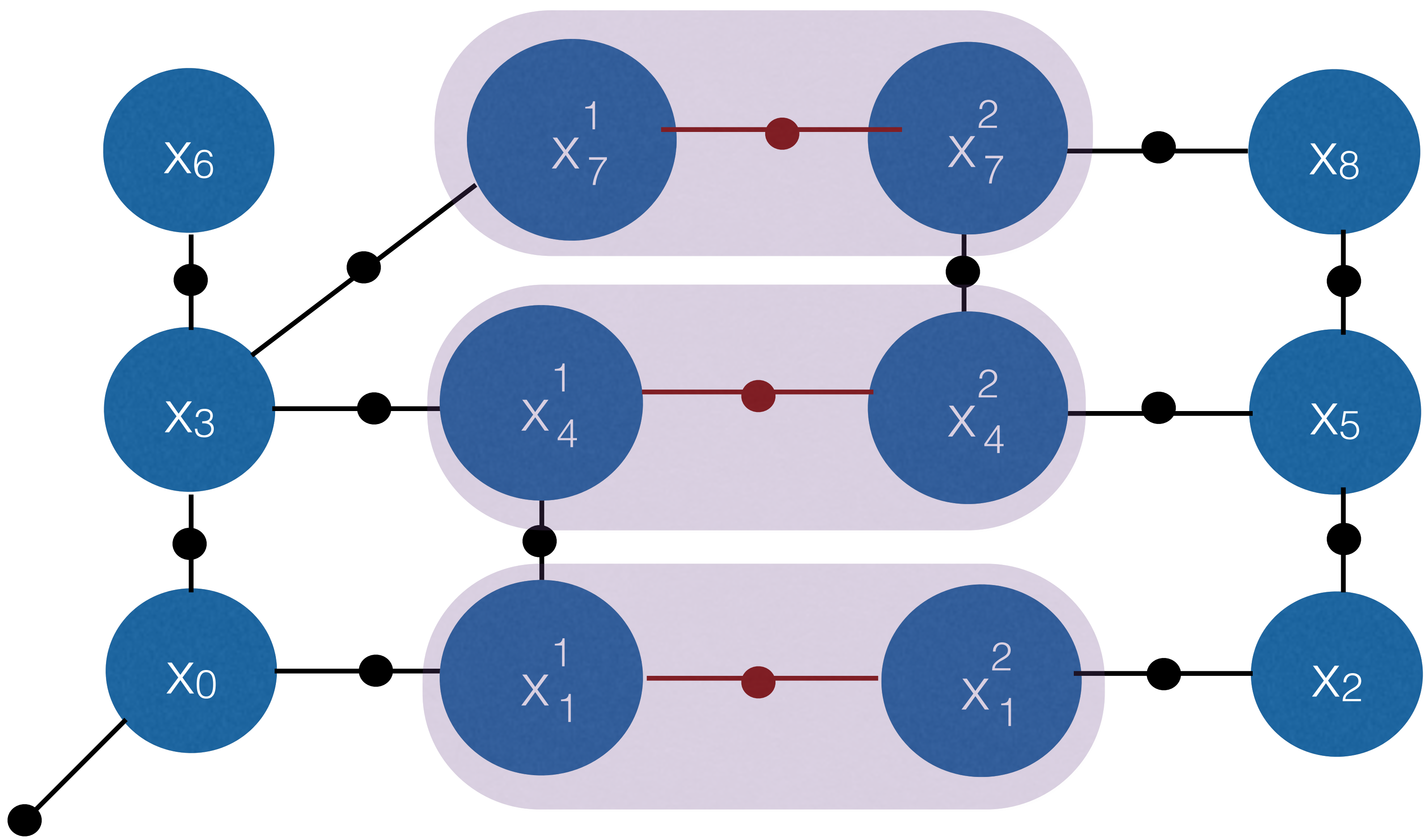


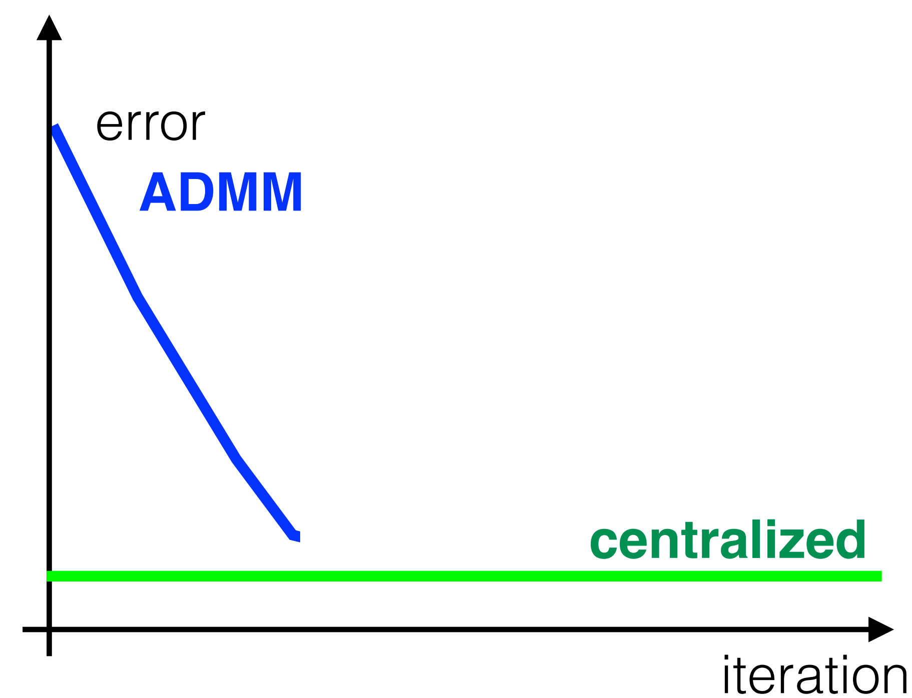


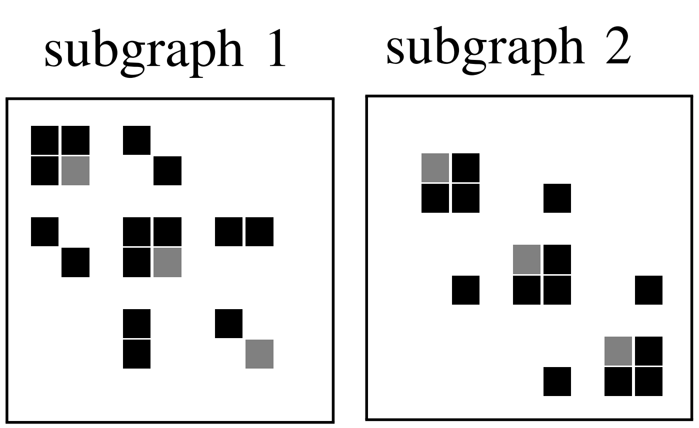
How ADMM works?

$$y_s\left(t+1\right) = y_s\left(t\right) + \sum_{s \in \mathcal{S}} \left[x_s^1 - x_s^2\right]$$

Update dual variables



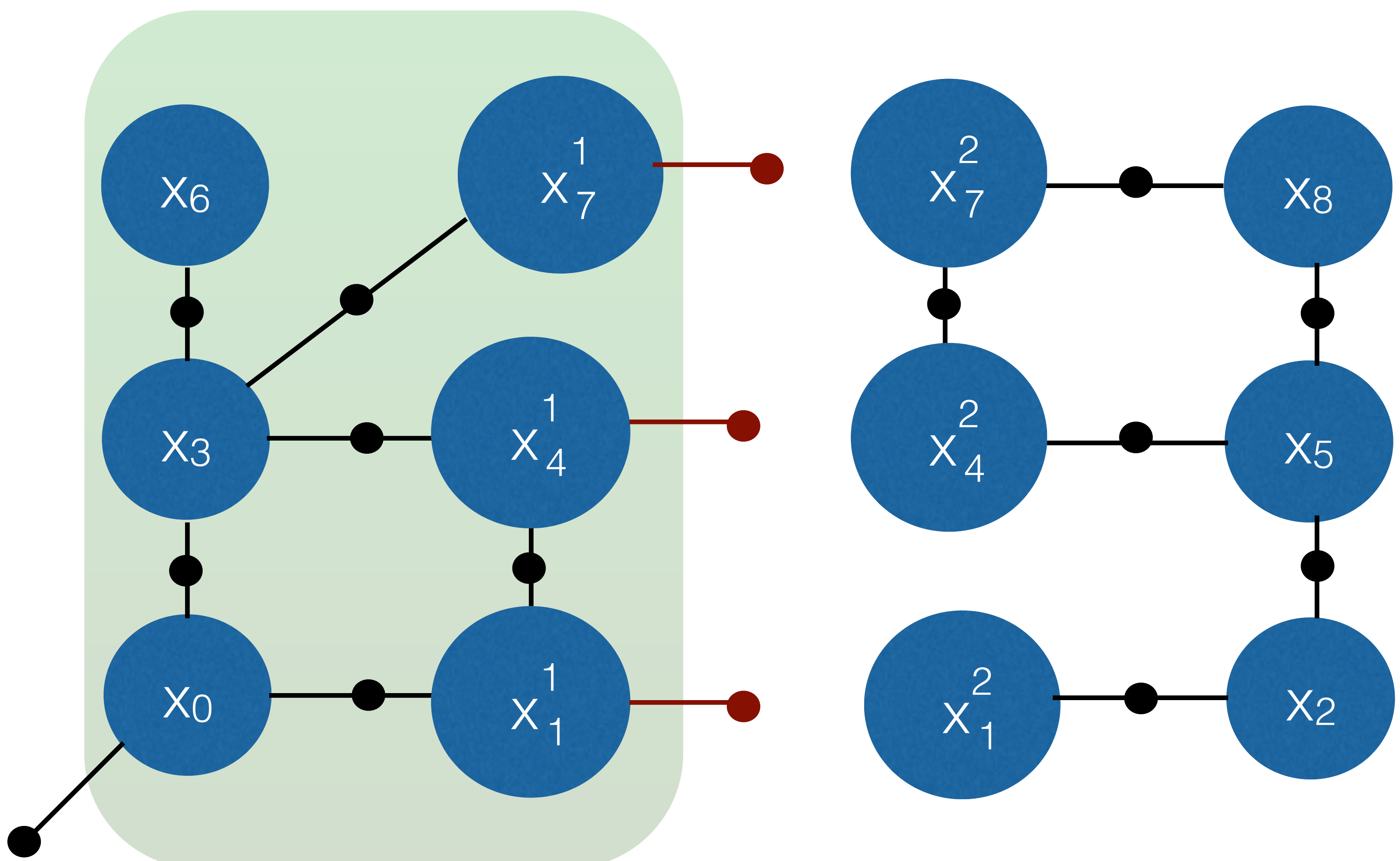


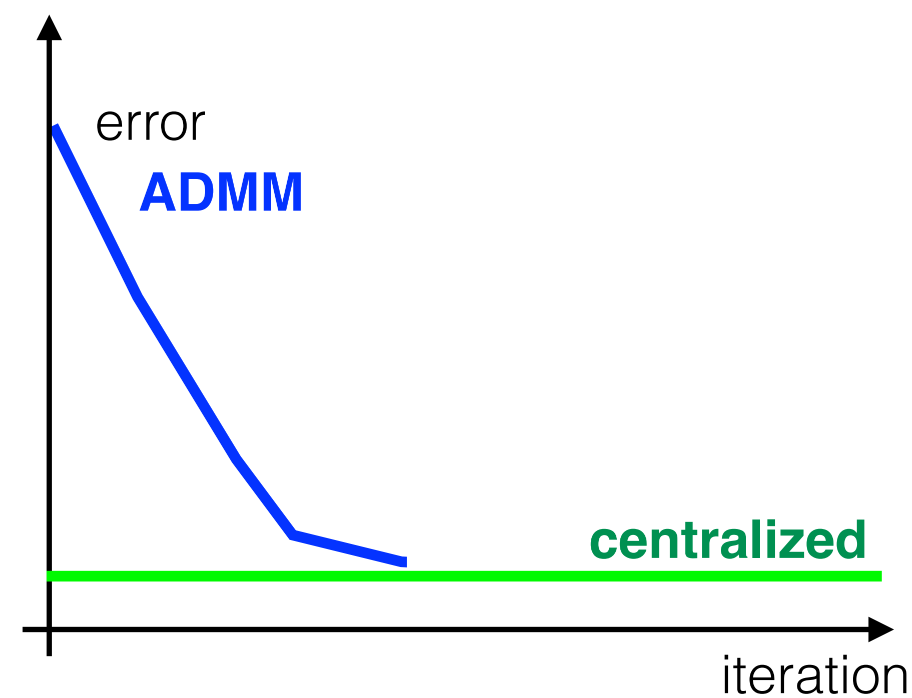


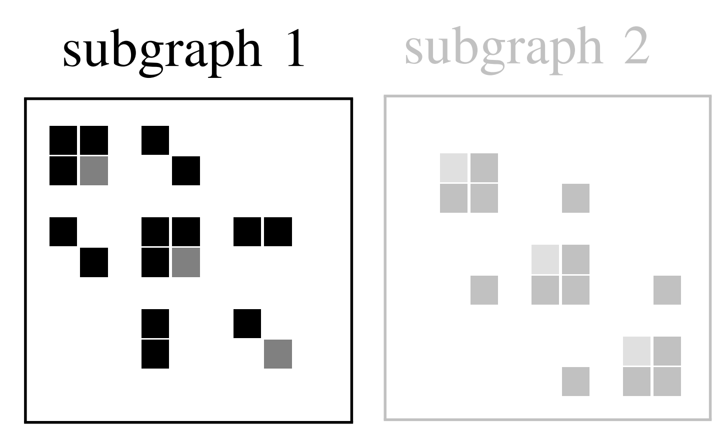
How ADMM works?

$$f\left(x^1,\mathcal{E}^1,\mathcal{P}^1\right) + f\left(x^2,\mathcal{E}^2,\mathcal{P}^2\right) + \sum_{s\in\mathcal{S}} \left[\frac{\rho}{2}\|x_s^1 - x_s^2\|^2 + y_s^T\left(x_s^1 - x_s^2\right)\right]$$

Optimize 1st Subgraph

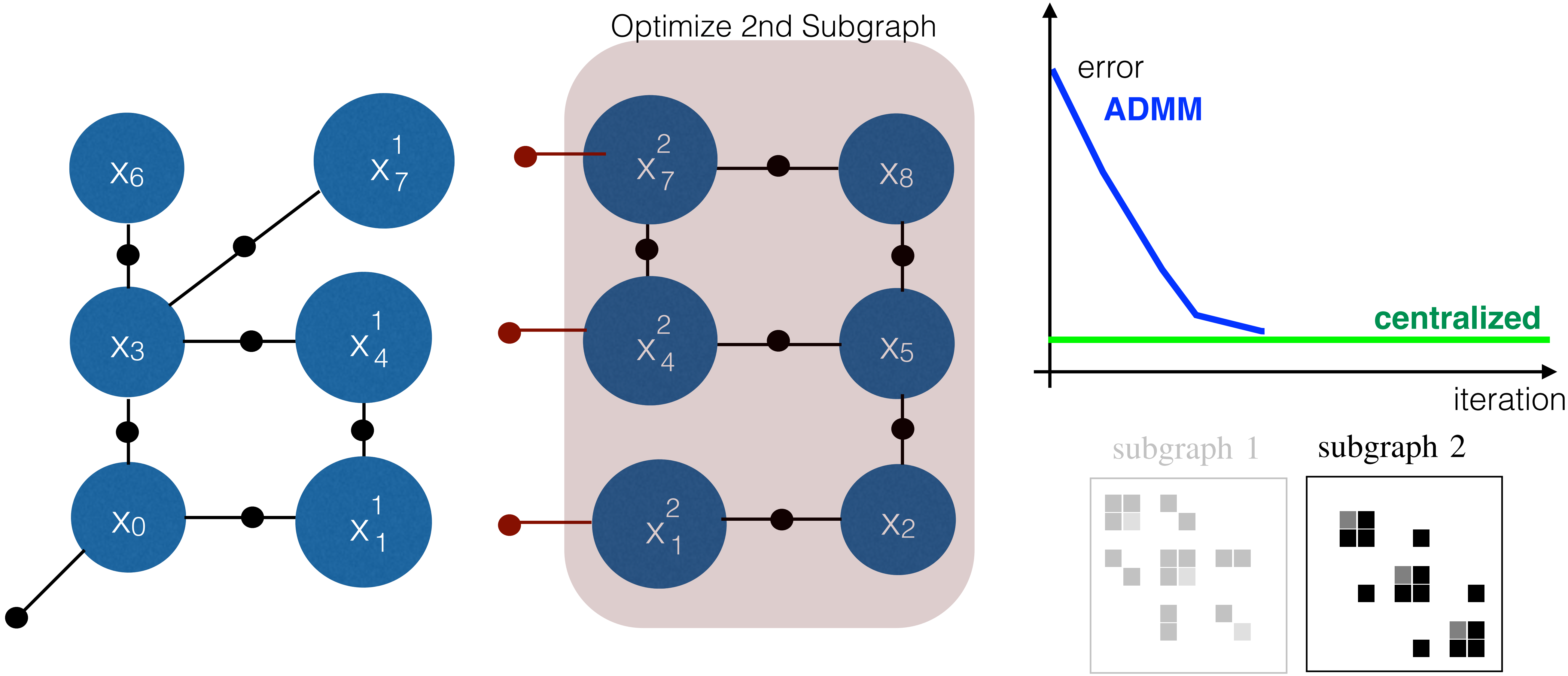






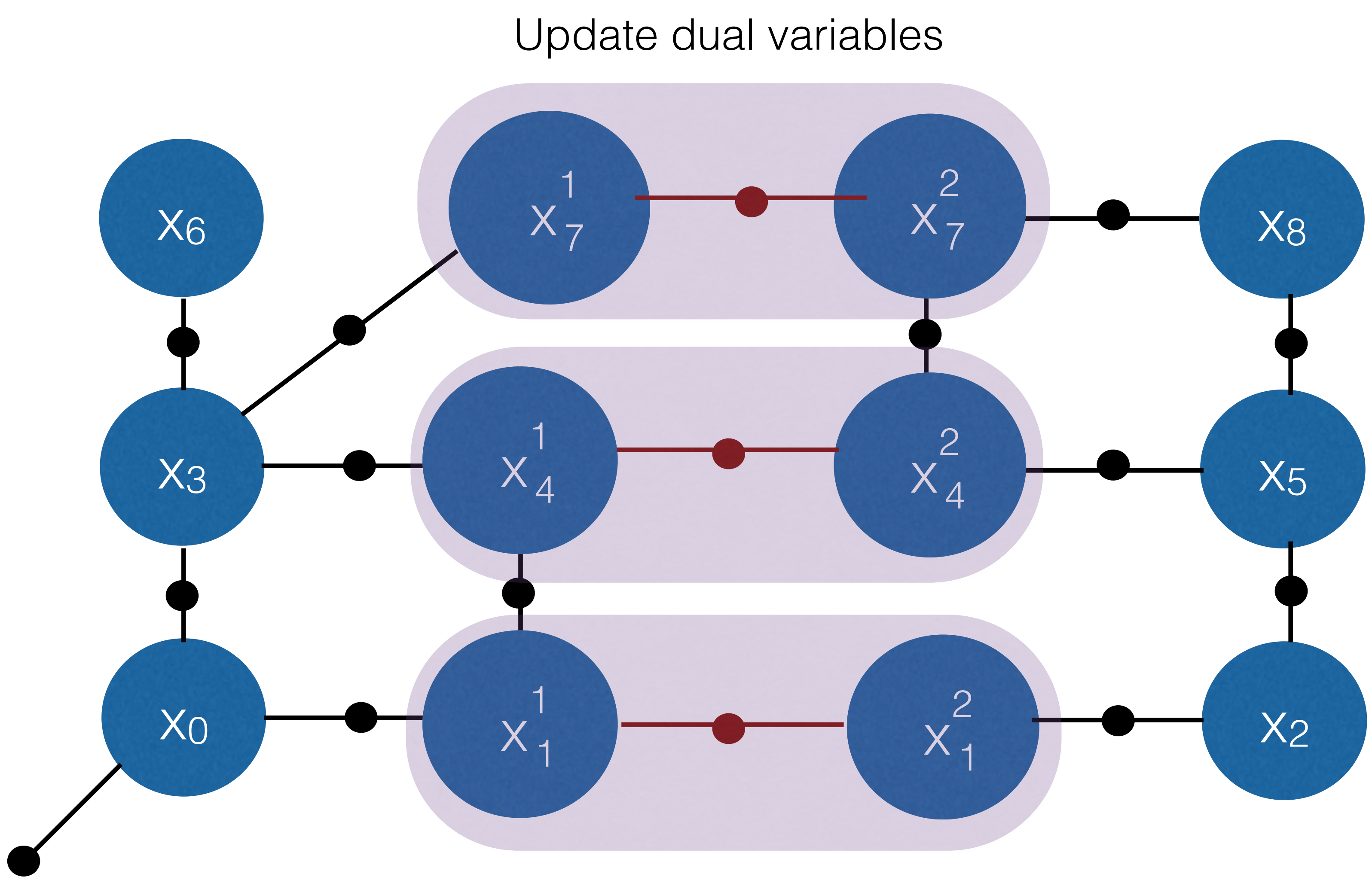
How ADMM works?

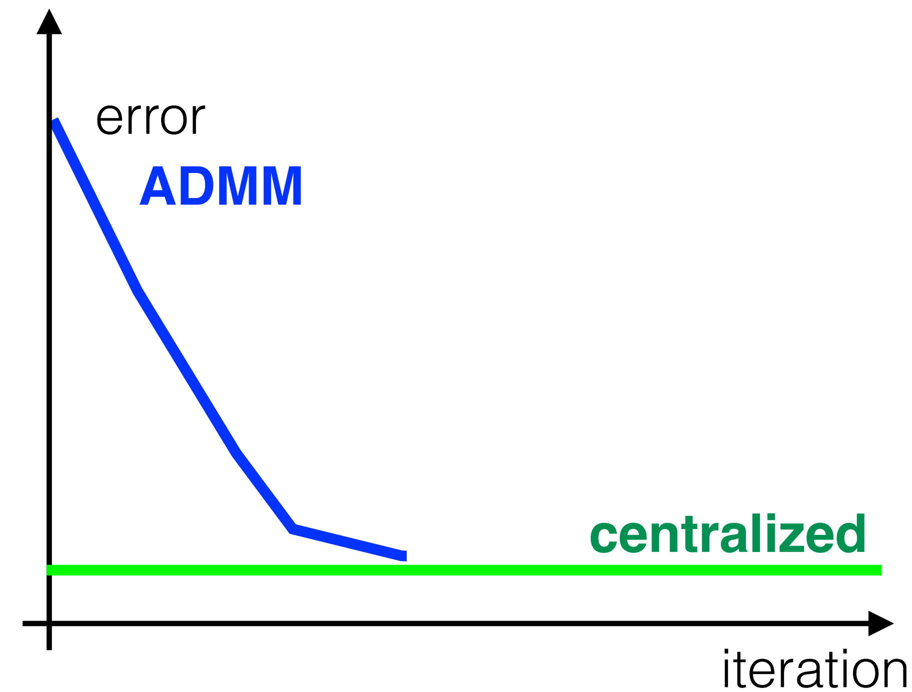
$$f\left(x^1,\mathcal{E}^1,\mathcal{P}^1\right)+\underline{f\left(x^2,\mathcal{E}^2,\mathcal{P}^2\right)}+\sum_{s\in\mathcal{S}}\left[\frac{\rho}{2}\|x_s^1-\underline{x_s^2}\|^2+y_s^T\left(x_s^1-\underline{x_s^2}\right)\right]$$

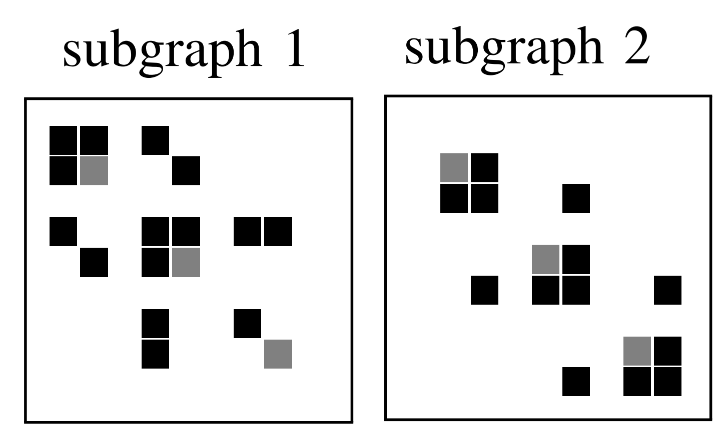


How ADMM works?

$$y_s\,(t+1) = y_s\,(t) + \sum_{s\in\mathcal{S}} \left[x_s^1 - x_s^2\right]$$



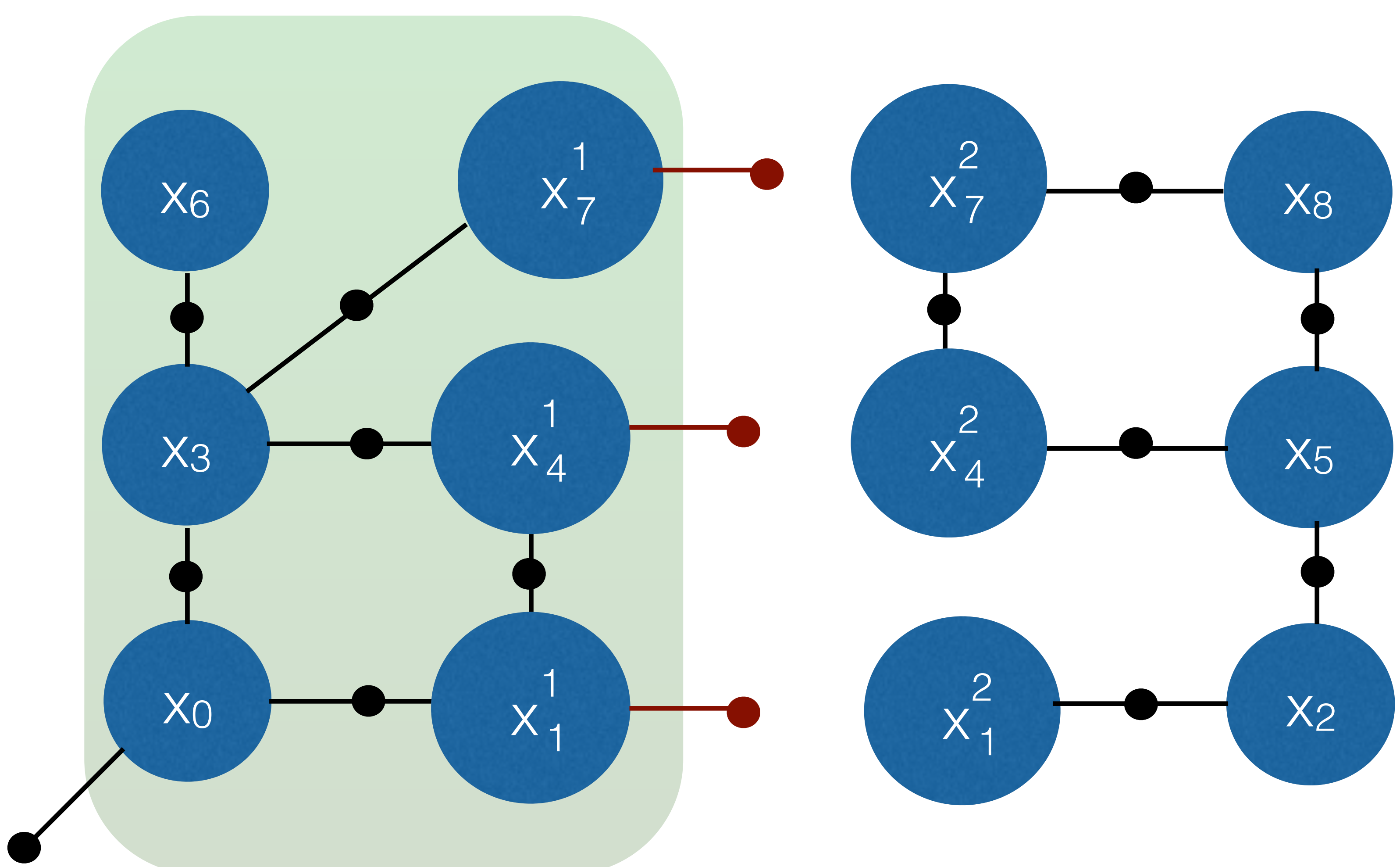


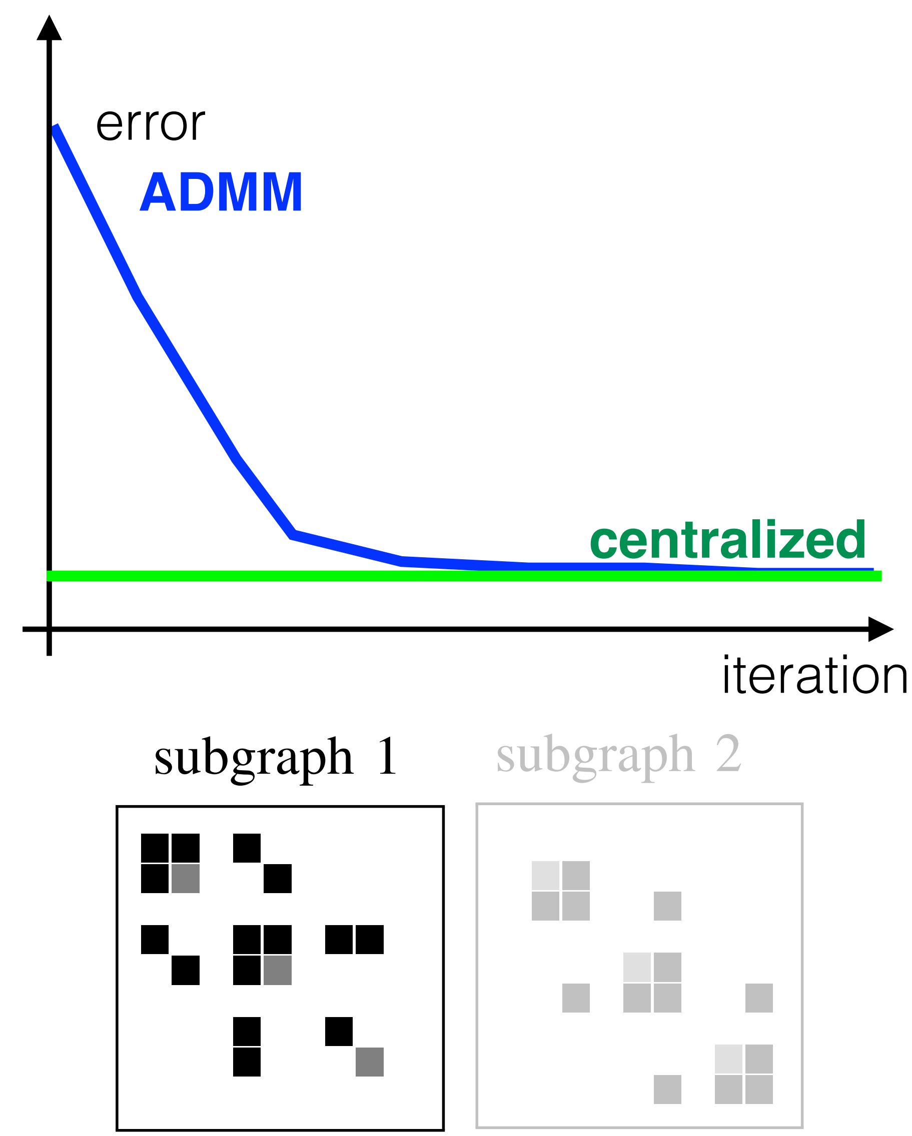


How ADMM works?

$$f\left(x^1, \mathcal{E}^1, \mathcal{P}^1\right) + f\left(x^2, \mathcal{E}^2, \mathcal{P}^2\right) + \sum_{s \in \mathcal{S}} \left[\frac{\rho}{2}\|x_s^1 - x_s^2\|^2 + y_s^T\left(x_s^1 - x_s^2\right)\right]$$

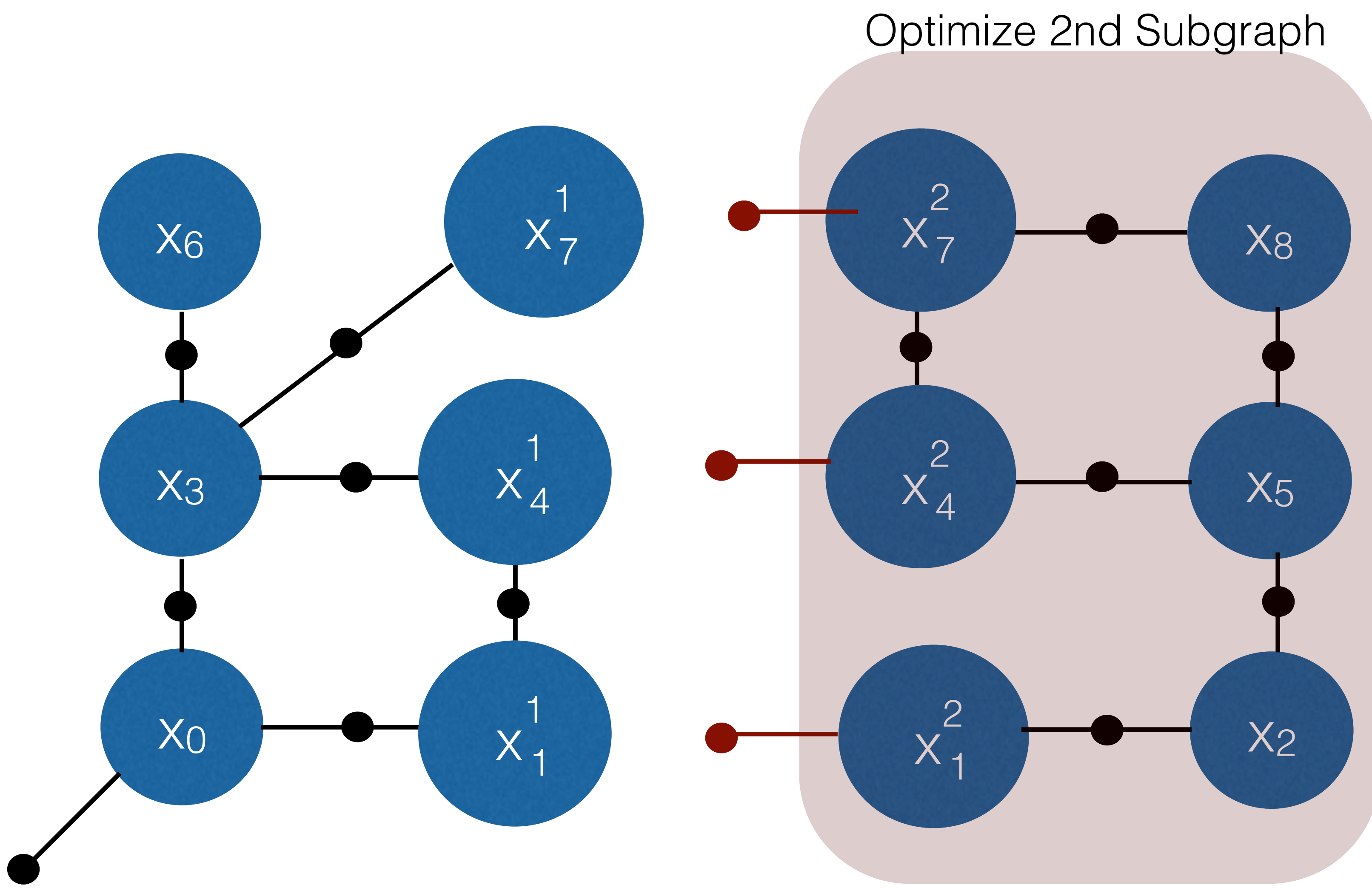
Optimize 1st Subgraph

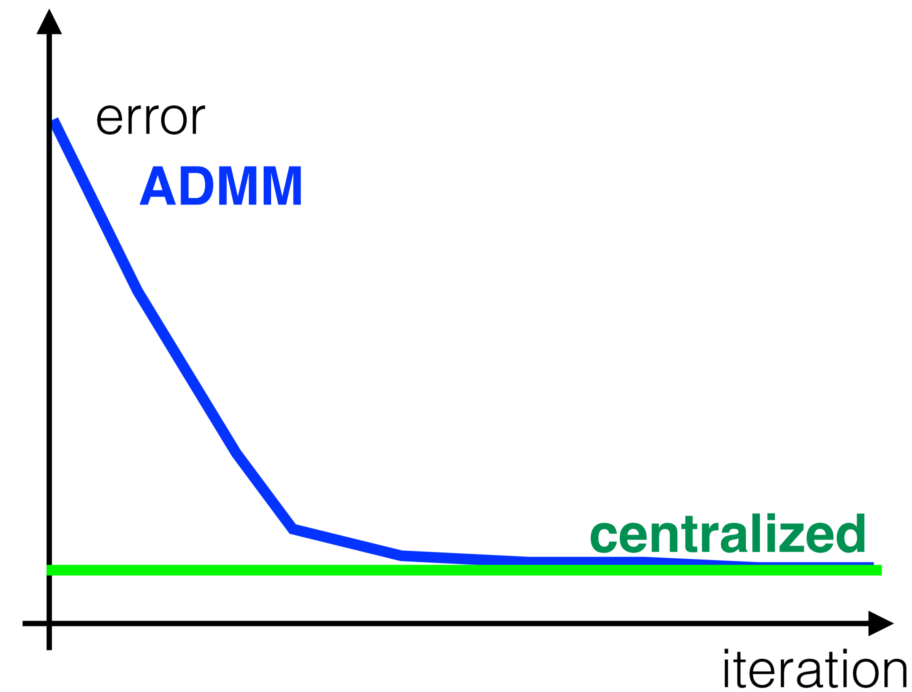


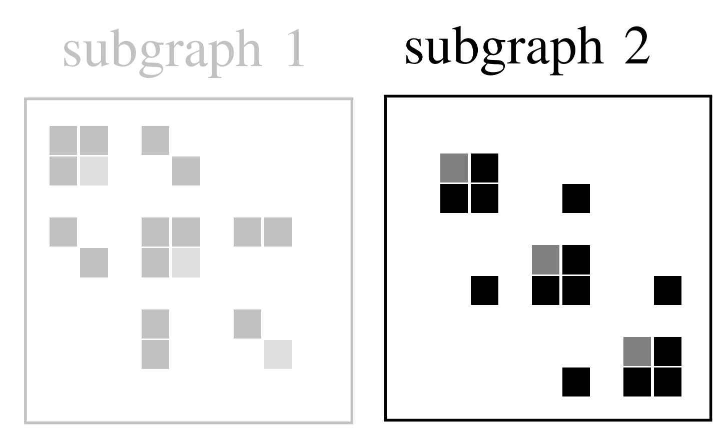


How ADMM works?

$$f\left(x^1,\mathcal{E}^1,\mathcal{P}^1\right)+\underline{f\left(x^2,\mathcal{E}^2,\mathcal{P}^2\right)}+\sum_{s\in\mathcal{S}}\left[\frac{\rho}{2}\|x_s^1-\underline{x_s^2}\|^2+y_s^T\left(x_s^1-\underline{x_s^2}\right)\right]$$



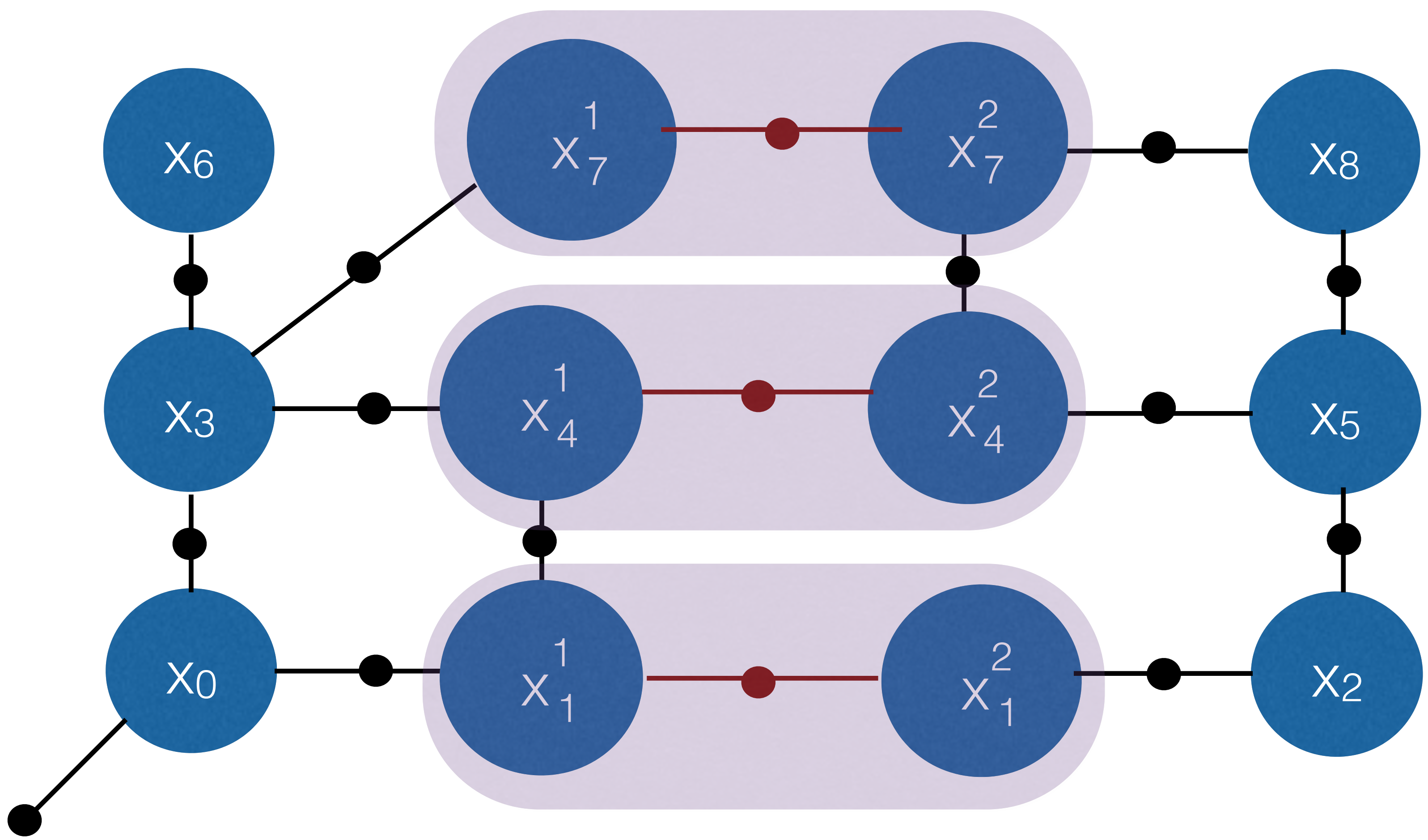


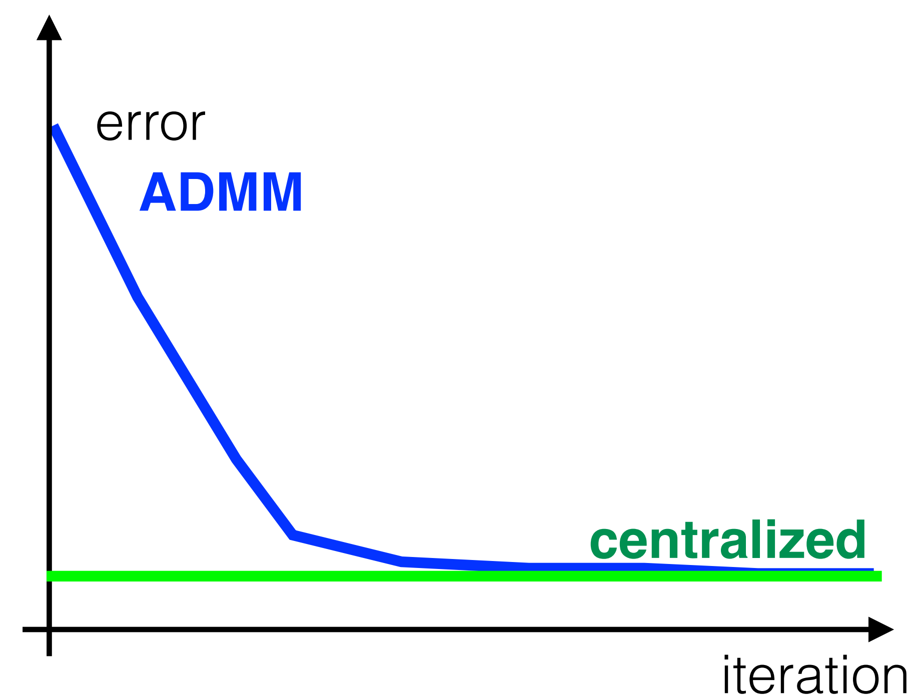


How ADMM works?

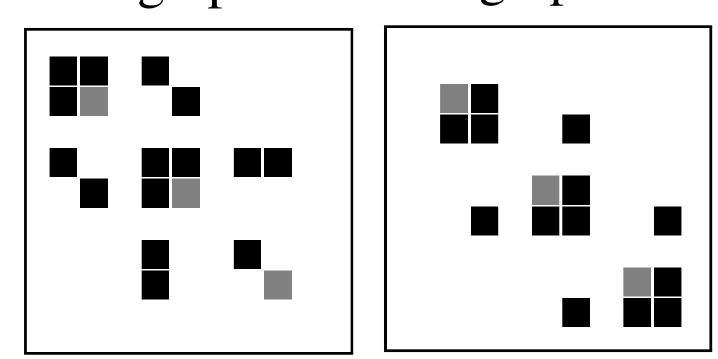
$$y_s(t+1) = y_s(t) + \sum_{s \in \mathcal{S}} \left[x_s^1 - x_s^2 \right]$$

Update dual variables





subgraph 1 subgraph 2



How ADMM works?

for K iterations

- optimize each subgraph

$$f\left(x^1, \mathcal{E}^1, \mathcal{P}^1\right) + f\left(x^2, \mathcal{E}^2, \mathcal{P}^2\right) + \sum_{s \in \mathcal{S}} \left[\frac{\rho}{2} \|x_s^1 - x_s^2\|^2 + y_s^T \left(x_s^1 - x_s^2\right)\right]$$

- update dual variables

$$y_s\left(t+1\right) = y_s\left(t\right) + \sum_{s \in \mathcal{S}} \left[x_s^1 - x_s^2\right]$$

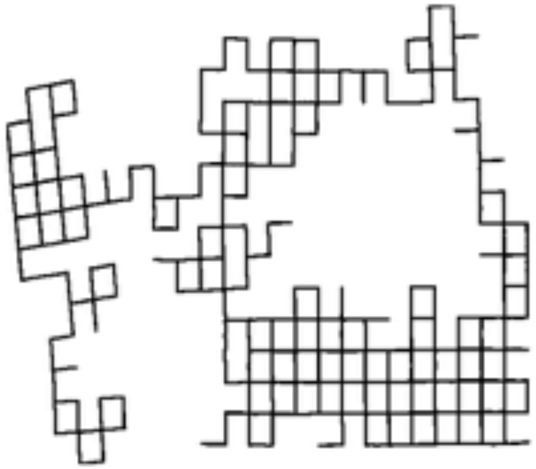
- break if stopping conditions are met
  (primal and dual residual less than a threshold)

Benchmark Datasets

AIS2Klink
15115 nodes,
16727 edges

ETHCampus
7065 nodes,
7427 edges

INTEL
1728 nodes,
2512 edges



M3500
3500 nodes,
5598 edges

CSAIL
1045 nodes,
1172 edges

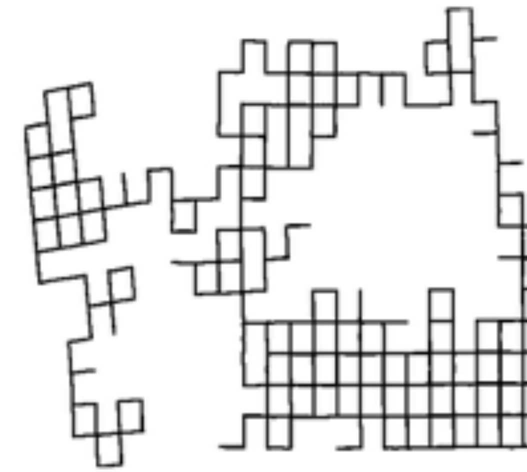
FR079
989 nodes,
1217 edges

Benchmark Datasets

AIS2Klink
15115 nodes,
16727 edges

ETHCampus
7065 nodes,
7427 edges

INTEL
1728 nodes,
2512 edges

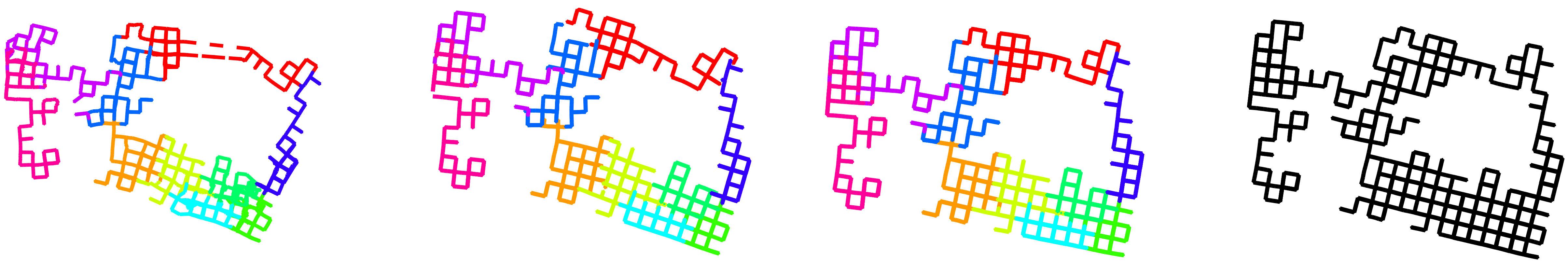


M3500
3500 nodes,
5598 edges

CSAIL
1045 nodes,
1172 edges

FR079
989 nodes,
1217 edges

Results (Just-in-Time Flavor)



Initial 1 Iteration 10 Iteration Centralized

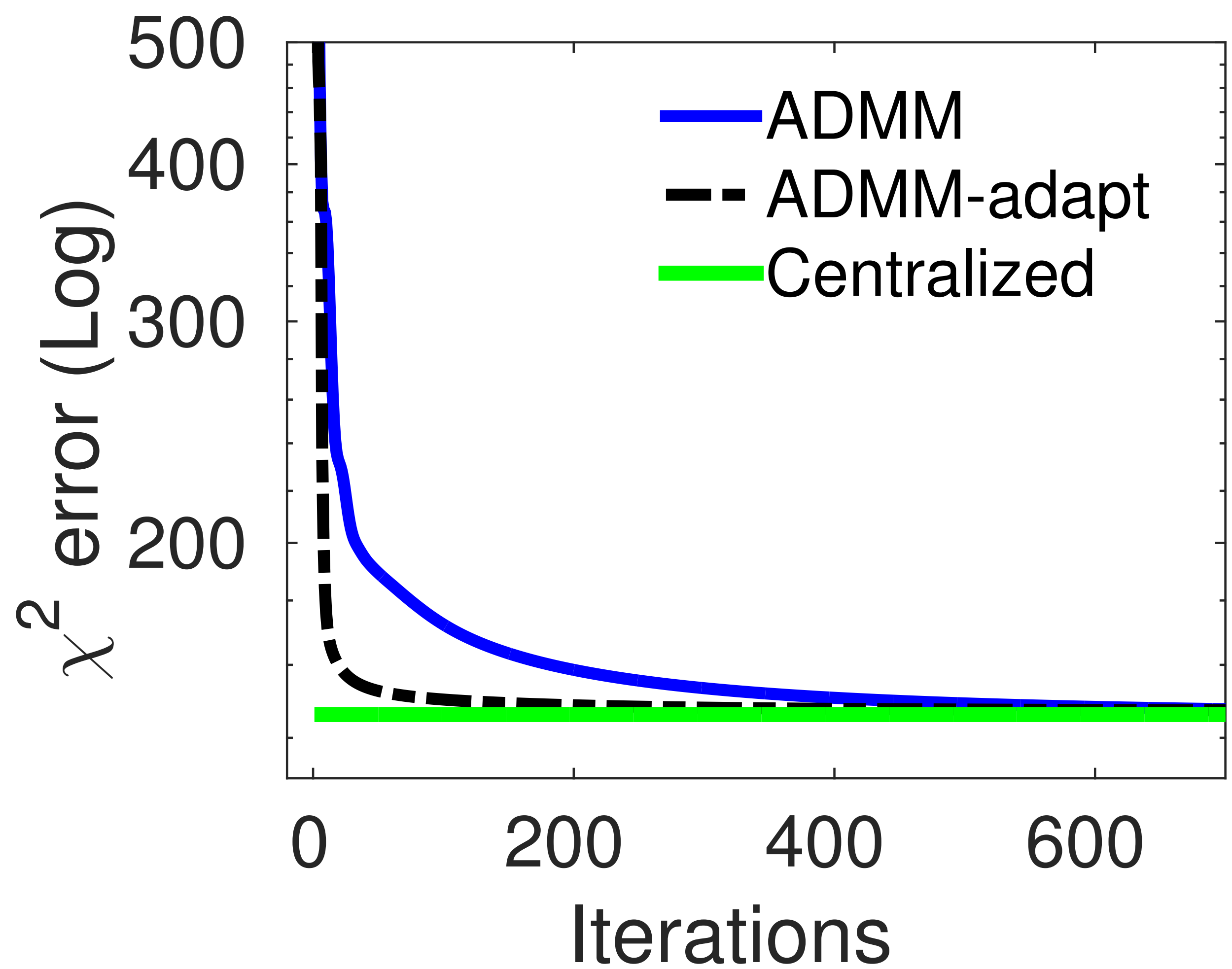
M3500 Dataset

Reaches a reasonable error in a few iterations and with additional CPU time it converges to the same accuracy as centralized solution.

Results (Convergence)

$$\min_{x^i \in \mathrm{SE}(2)} f(x^i, \mathcal{E}^i, \mathcal{P}^i) + \sum_{s \in \mathcal{S}_i} \frac{\rho}{2} \left\| \mathrm{Log}\left((x_s^i)^{-1} x_s^j(t) \right) + \frac{y_s}{\boxed{\rho}} \right\|^2$$

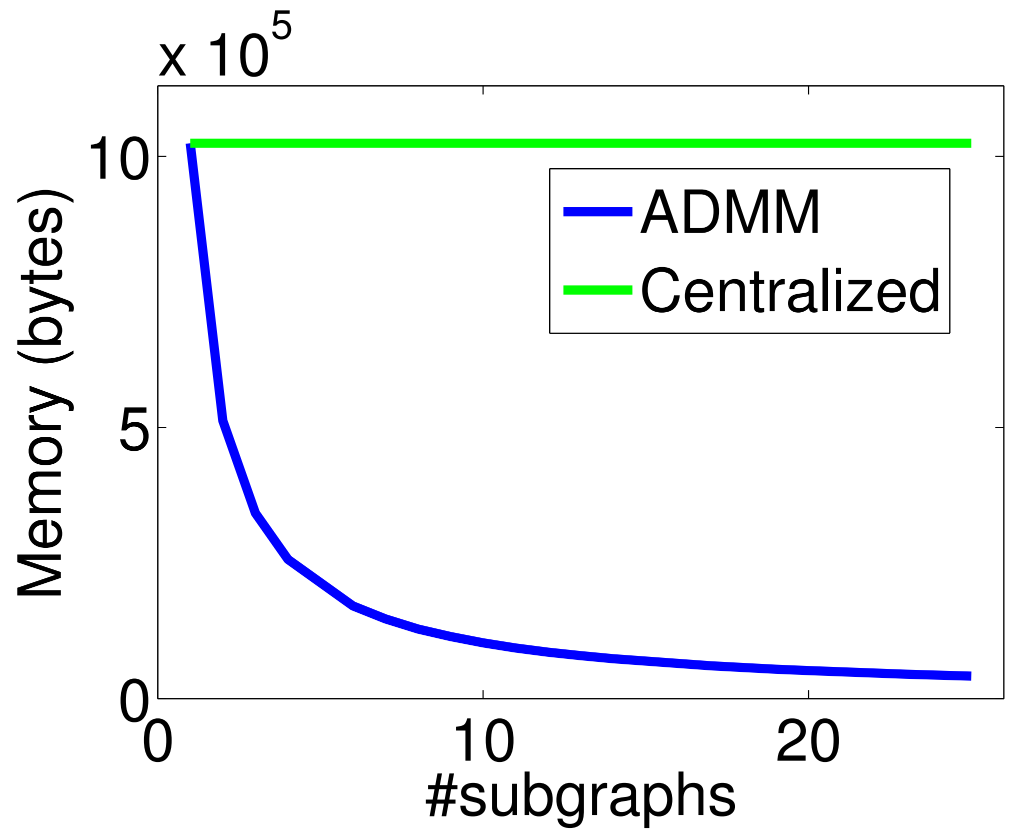
Parameter

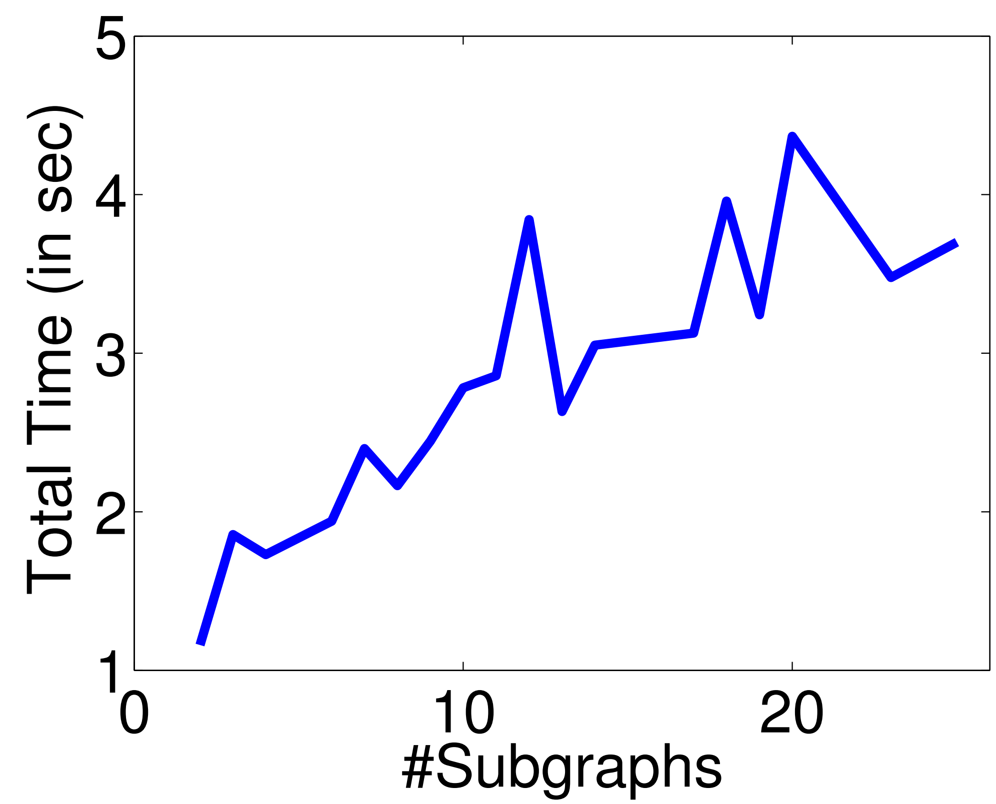


M3500 Dataset

Results (Memory Efficient)

Terminate when cost reaches within 10% of the centralized cost





M3500 Dataset

We can easily accommodate systems with small memory by using more subgraphs.

Results (State of Art)

Comparison against the state of art

Dataset	Method	χ^2 error	time [s]
AIS2Klinik	ADMM-adapt	**194.9**	84.70
	SMF	471.0	0.86
	T2-NOREL	108977.8	1.00
	HogMan	647.0	15.53
	TSAM2	**172.8**	2.85
ETHCampus	ADMM-adapt	**28.6**	24.64
	SMF	38.9	0.36
	T2-NOREL	22457.2	0.50
	HogMan	79.3	2.55
	TSAM2	**25.0**	1.15
INTEL	ADMM-adapt	54.6	8.38
	SMF	**53.3**	0.11
	T2-NOREL	69.0	0.08
	HogMan	134.7	0.78
	TSAM2	**45.0**	0.18
M3500	ADMM-adapt	**148.6**	23.33
	SMF	287.1	0.35
	T2-NOREL	733.8	0.21
	HogMan	521.9	3.25
	TSAM2	**146.1**	0.54

Results (State of Art)

Comparison against the state of art

Dataset	Method	χ^2 error	time [s]
AIS2Klinik	ADMM-adapt	**194.9**	84.70
	SMF	471.0	0.86
	T2-NOREL	108977.8	1.00
	HogMan	647.0	15.53
	TSAM2	**172.8**	2.85
ETHCampus	ADMM-adapt	**28.6**	24.64
	SMF	38.9	0.36
	T2-NOREL	22457.2	0.50
	HogMan	79.3	2.55
	TSAM2	**25.0**	1.15
INTEL	ADMM-adapt	54.6	8.38
	SMF	**53.3**	0.11
	T2-NOREL	69.0	0.08
	HogMan	134.7	0.78
	TSAM2	**45.0**	0.18
M3500	ADMM-adapt	**148.6**	23.33
	SMF	287.1	0.35
	T2-NOREL	733.8	0.21
	HogMan	521.9	3.25
	TSAM2	**146.1**	0.54

Our approach is:

- **more accurate than SMF (no approximations)**

Results (State of Art)

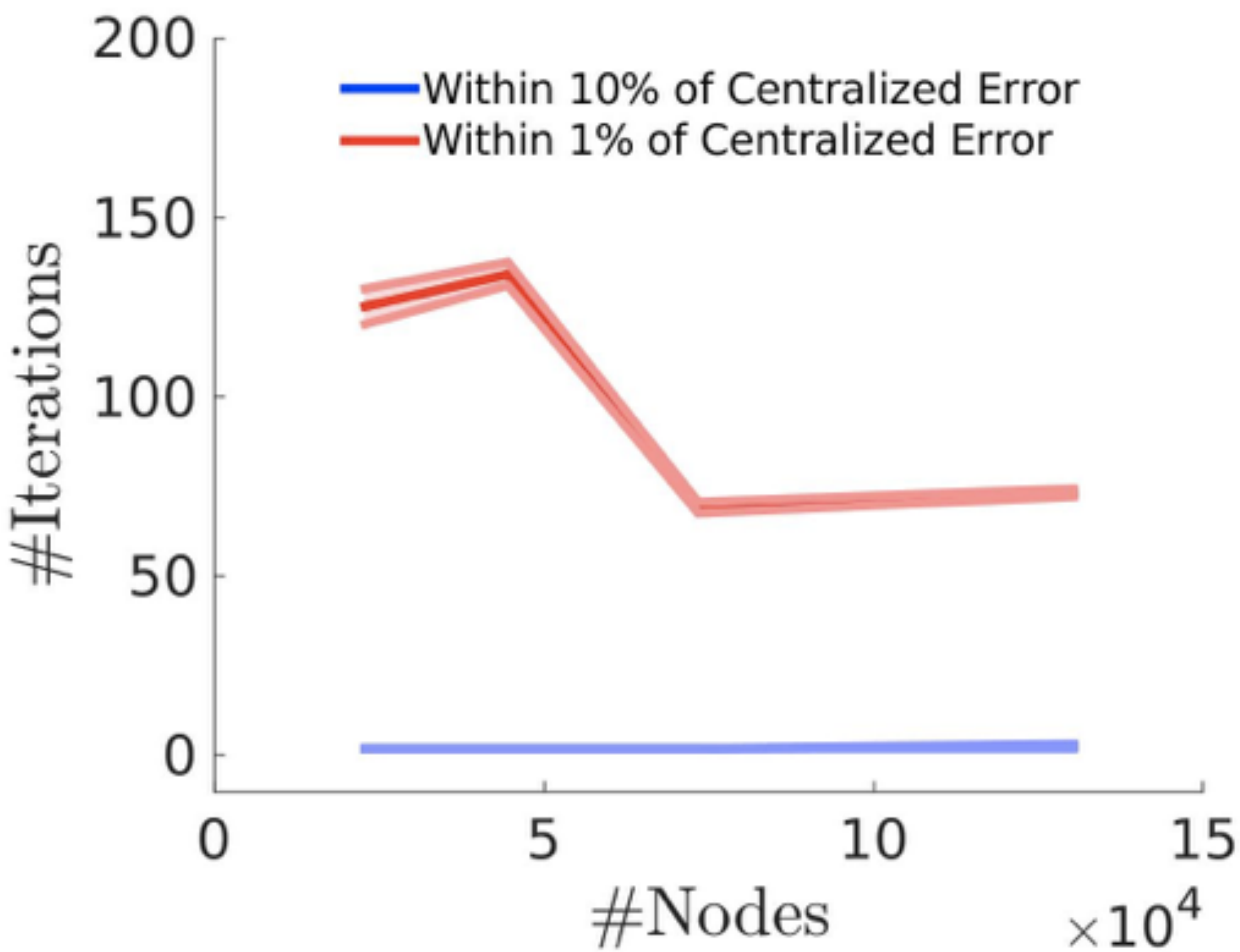
Comparison against the state of art

Dataset	Method	χ^2 error	time [s]
AIS2Klinik	ADMM-adapt	**194.9**	84.70
	SMF	471.0	0.86
	T2-NOREL	108977.8	1.00
	HogMan	647.0	15.53
	TSAM2	**172.8**	2.85
ETHCampus	ADMM-adapt	**28.6**	24.64
	SMF	38.9	0.36
	T2-NOREL	22457.2	0.50
	HogMan	79.3	2.55
	TSAM2	**25.0**	1.15
INTEL	ADMM-adapt	54.6	8.38
	SMF	**53.3**	0.11
	T2-NOREL	69.0	0.08
	HogMan	134.7	0.78
	TSAM2	**45.0**	0.18
M3500	ADMM-adapt	**148.6**	23.33
	SMF	287.1	0.35
	T2-NOREL	733.8	0.21
	HogMan	521.9	3.25
	TSAM2	**146.1**	0.54

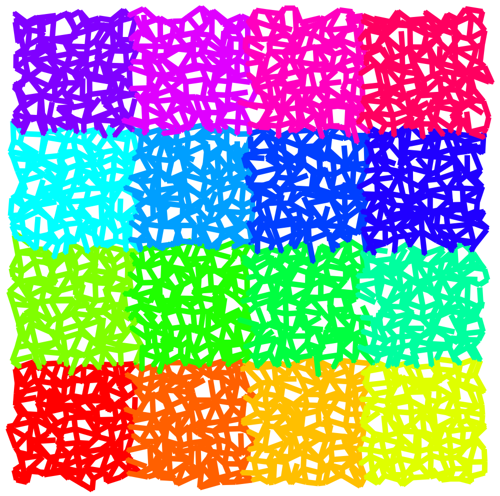
Our approach is:

- **more accurate than SMF (no approximations)**
- **more memory efficient than TSAM2 (sparse)**

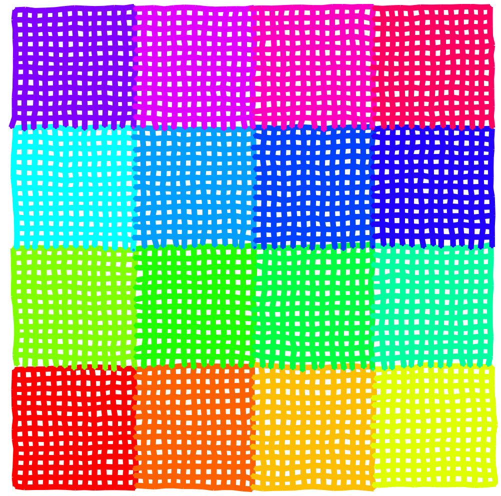
Results (Scalability)



Over 10 Monte-Carlo Samples



Initial



After 20 iterations

Conclusions

- **Exactly sparse**, does not require joint optimization over separators.
- **Accurate**, does not use approximations (eg. sparsification).
- **Memory Efficient**, can accommodate small memory systems by increasing the number of subgraphs.
- **Straightforward implementation**, does not require hierarchical schemes or linearization point bookkeeping.

Limitations and Future Work

- **Slower than other approaches** due to sequential optimization of subgraphs and slow ascent of dual variables
  - Future work: **Parallelization** of ADMM iterations.

Other ideas:

- **Extending** it to more general factor graphs (eg. landmark-SLAM).
- **Explore** applications of ADMM in Multi-robot setup.

Georgia Tech Institute for Robotics and Intelligent Machines

LIDS MIT

Thank you!

For further information, contact:
Siddharth Choudhary
siddharth.choudhary@gatech.edu

We wish to thank Gian Diego Tipaldi and Benjamin Suger for sharing the datasets ETHCampus and AIS2Klinik, and for authorizing the use of the results. We gratefully acknowledge reviewers for the helpful comments.

This work was partially funded by the ARL MAST CTA Project 1436607 “Autonomous Multifunctional Mobile Microsystems” and by the National Science Foundation Award 11115678 “RI: Small: Ultra-Sparsifiers for Fast and Scalable Mapping and 3D Reconstruction on Mobile Robots”.

ARL

NSF