A Decomposition and Coordination Approach for Unit Commitment with Sequential and Parallel Implementations

Niranjan Raghunathan, Mikhail Bragin, Bing Yan, Peter B. Luh
Electrical & Computer Engineering
University of Connecticut

Khosrow Moslehi, Yaowen Yu, Xiaoming Feng, Chien-Ning Yu, Chia-Chun Tsai
ABB

June 26, 2019

Outline

- Introduction
- Problem formulation
- A novel decomposition and coordination algorithm
  - Foundation: Sequential Surrogate Absolute-Value Lagrangian Relaxation (SAVLR) + Branch-and-Cut (B&C)
  - Further improvements
- Numerical testing results
- Parallel SAVLR + B&C and testing results
- Concluding remarks

Introduction - UC

- Unit commitment (UC) problems are becoming increasingly larger in size and complexity, such as
  - Shorter time intervals and longer horizons
    - Maintain system reliability by having more control over resources
  - Various system- and area-level reserve requirements
    - Maintain system reliability
  - Discrete variables make problem combinatorial
    - Exponential growth in complexity as problem sizes increase

Introduction - UC

- Unit commitment (UC) problems are becoming increasingly larger in size and complexity, such as
  - Shorter time intervals and longer horizons
    - Maintain system reliability by having more control over resources
  - Various system- and area-level reserve requirements
    - Maintain system reliability
  - Discrete variables make problem combinatorial
    - Exponential growth in complexity as problem sizes increase
- Obtaining high-quality solutions to such problems is crucial for efficient and fair operation of large power systems
- Widely used branch and cut (B&C) may not obtain quality solutions in a reasonable amount of time

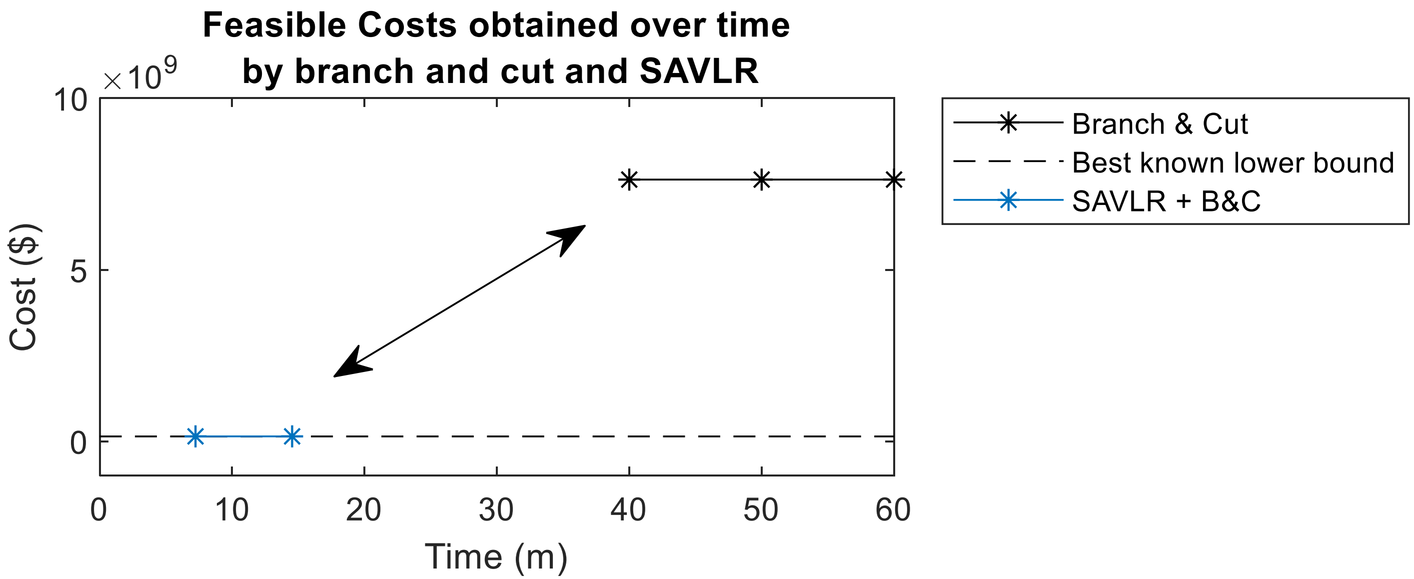
Introduction – Our work

- In our work, a multiple-hour unit commitment problem is created based on the publically available Polish system (usually used for power flow) with the following characteristics:
  - 15-minute time intervals for 12 hours
  - System- and area-level reserve requirements
  - Transmission capacity constraints
  - Transmission capacity and reserves modeled by soft constraints
    - Penalize constraint violations with predetermined penalties

Introduction – Our work

- In our work, a multiple-hour unit commitment problem is created based on the publically available Polish system (usually used for power flow) with the following characteristics:
  - 15-minute time intervals for 12 hours
  - System- and area-level reserve requirements
  - Transmission capacity constraints
  - Transmission capacity and reserves modeled by soft constraints
    - Penalize constraint violations with predetermined penalties
- We present a novel decomposition and coordination algorithm
  - Exploit separability and accelerate convergence by penalties
  - Improve convergence through selective relaxation of constraints
  - Improve computational efficiency
  - Further speed up by efficient parallelization

Introduction – A glance of testing results



- B&C does not find a quality solution within an hour *
- SAVLR + B&C finds a near- optimal solution within 10 min

*: A tough testing case is created by UConn. The testing is performed on MATLAB R2018a and commercial solver IBM ILOG CPLEX Optimization Studio V 12.8.0.0 on a PC with 2.90GHz Intel Core(TM) i7 CPU and 32G RAM.

Problem Formulation

$$\min f\left(p,u,x,r,srr,stsc\right)=$$

$$\sum_{t\in T}\left(\sum_{j\in G}\left(C_j\left(p_j\left(t\right)\right)+C_j^{\mathrm{SU}}u_j\left(t\right)+C_j^{\mathrm{NL}}x_j\left(t\right)+\sum_{m\in M}C_{m,j}^{\mathrm{R}}r_{m,j}\left(t\right)\right)\right.$$

Reserve penalty — Transmission capacity penalty

$$\left.+\sum_{a\in A}\sum_{n\in NR}C_{n,a}^{\mathrm{RP}}srr_{n,a}\left(t\right)+\sum_{l\in L}C_l^{\mathrm{TC}}\left(stsc_l^{+}\left(t\right)+stsc_l^{-}\left(t\right)\right)\right)$$

- subject to unit, area, and system level constraints
- Soft reserve constraints:

$$\sum_{j\in G}\left(r_{1,j}^{k}\left(t\right)\times \mathrm{pa}_{j,a}\right)+srr_{1,a}\left(t\right)\geq \mathrm{RR}_{1,a}\left(t\right)$$

Penalty variable allows constraint to be violated

- Why soft constraints?
  - Ensure technical feasibility when the original problem is infeasible [1]
  - Simplify coordination of our algorithm by not relaxing these constraints

1. Y. M. Al-Abdullah, A. Salloum, K. W. Hedman and V. Vittal, "Analyzing the Impacts of Constraint Relaxation Practices in Electric Energy Markets," in *IEEE Transactions on Power Systems*, vol. 31, no. 4, pp. 2566-2577, July 2016.

Foundations of our novel algorithm

- Decomposition and coordination - Lagrangian Relaxation (LR)
  - A "dual" approach (prices as decision variables)
- Major difficulties with discrete variables:
  - Solving all subproblems is time consuming
  - Multipliers suffer from major zigzagging
  - Require the optimal dual value q^*

Foundations of our novel algorithm

- Decomposition and coordination - Lagrangian Relaxation (LR)
  - A "dual" approach (prices as decision variables)
- Major difficulties with discrete variables:
  - Solving all subproblems is time consuming
  - Multipliers suffer from major zigzagging
  - Require the optimal dual value q^*
- Surrogate Lagrangian Relaxation (SLR): overcame the above
  - Surrogate optimality condition
    - Ensure directions point toward the optimal multipliers
    - Obtain smoother directions with less effort (1 subproblem)
  - Stepsizing rule without requiring q^* → Guarantee convergence

Foundations of our novel algorithm

- Decomposition and coordination - Lagrangian Relaxation (LR)
  - A "dual" approach (prices as decision variables)
- Major difficulties with discrete variables:
  - Solving all subproblems is time consuming
  - Multipliers suffer from major zigzagging
  - Require the optimal dual value q^*
- Surrogate Lagrangian Relaxation (SLR): overcame the above
  - Surrogate optimality condition
    - Ensure directions point toward the optimal multipliers
    - Obtain smoother directions with less effort (1 subproblem)
  - Stepsizing rule without requiring q^* → Guarantee convergence
- Surrogate Absolute-Value Lagrangian Relaxation (SAVLR):
  - Accelerate convergence by using absolute value penalties

Further improvements

- Improve convergence through selective relaxation of constraints
  - Numerous system-wide coupling constraints ~ slow coordination
  - Only relax the demand constraints
  - Penalize soft coupling constraints, not relax
  - Fewer multipliers → faster convergence to optimal values

Further improvements

- Improve convergence through selective relaxation of constraints
  - Numerous system-wide coupling constraints ~ slow coordination
  - Only relax the demand constraints
  - Penalize soft coupling constraints, not relax
  - Fewer multipliers → faster convergence to optimal values
- Improve computational efficiency
  - Build only varying components of subproblem models at each iteration

Further improvements

- Improve convergence through selective relaxation of constraints
  - Numerous system-wide coupling constraints ~ slow coordination
  - Only relax the demand constraints
  - Penalize soft coupling constraints, not relax
  - Fewer multipliers → faster convergence to optimal values
- Improve computational efficiency
  - Build only varying components of subproblem models at each iteration
- Further speed up by efficient parallelization
  - Solve a few subproblems at a time in parallel

Relaxed problem with absolute value penalties

- Relax the demand constraints, and penalize the constraint violation by absolute value functions
- Penalize soft coupling constraints, i.e., reserve requirements and transmission capacity constraints

$$\min L_c\left(p,u,x,r,srr,stsc\right)=$$

$$\sum_{t\in T}\left(\sum_{j\in G}\left(C_j\left(p_j\left(t\right)\right)+C_j^{\mathrm{SU}}u_j\left(t\right)+C_j^{\mathrm{NL}}x_j\left(t\right)+\sum_{m\in M}C_{m,j}^{\mathrm{R}}r_{m,j}\left(t\right)\right)\right.$$

$$+\sum_{a\in A}\sum_{n\in NR}C_{n,a}^{\mathrm{RP}}srr_{n,a}\left(t\right)+\sum_{l\in L}C_l^{\mathrm{TC}}\left(stsc_l^{+}\left(t\right)+stsc_l^{-}\left(t\right)\right)$$

$$\left.+\lambda\left(t\right)\left(\sum_{i\in I}P_i^{\mathrm{D}}\left(t\right)-\sum_{j\in G}p_j\left(t\right)\right)\right)+\frac{c}{2}\left\|\left(\sum_{i\in I}^{N}P_i^{\mathrm{D}}\left(t\right)-\sum_{j\in G}^{I}p_j\left(t\right)\right)\right\|$$

Relaxed demand

Absolute value penalties for constraint violations accelerate convergence

Subproblem formulation

$$\min L_{c^k}\left(p^k_{j\in G_a}, p^{k-1}_{j\notin G_a}, u^k_{j\in G_a}, x^k_{j\in G_a}, r^k_{j\in G_a}, srr^k, stsc^{+,k}, stsc^{-,k}\right)=$$

$$\sum_{t\in T}\left(\sum_{j:j\in G_a} C_j p_j^k(t)+\sum_{j:j\in G_a} C_j^{SU} u_j^k(t)+\sum_{j:j\in G_a} C_j^{NL} x_j^k(t)+\sum_{j:j\in G_a}\sum_{m\in M} C^R_{m,j} r^k_{m,j}(t)+\sum_{n\in NR} C^P_{n,a} srr^k_{n,a}(t)\right.$$

$$\left.+\sum_{l\in L} C_l^{TP}\left(stsc_l^{+,k}(t)+stsc_l^{-,k}(t)\right)+\lambda^k(t)\left(y^{D+,k}(t)-y^{D-,k}(t)\right)+\frac{c^k}{2}\left(y^{D+,k}(t)+y^{D-,k}(t)\right)\right)$$

Linearized absolute value penalties

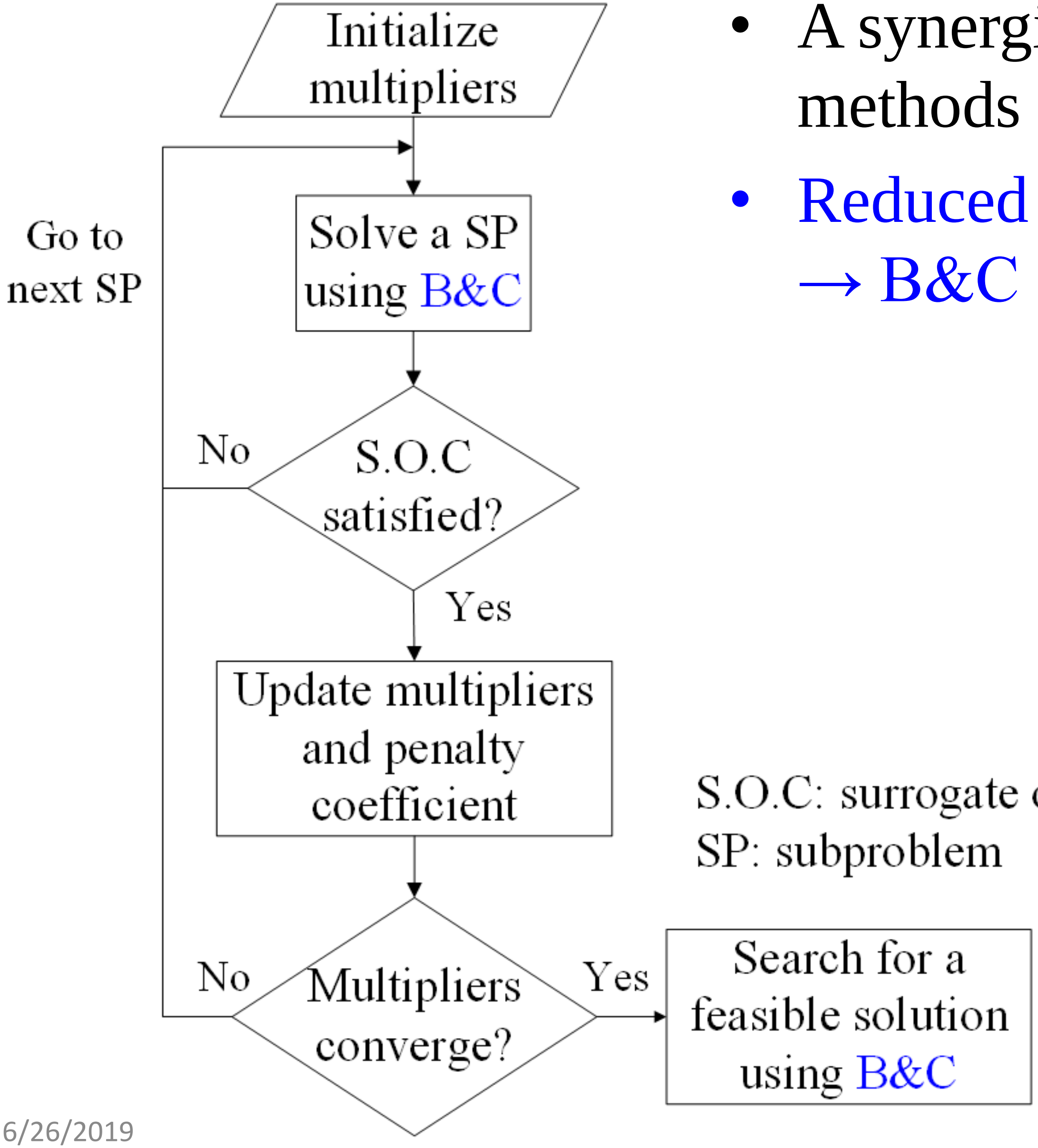
subject to:

$$y^{D+,k}(t)-y^{D-,k}(t)=\sum_{i\in I} P_i^D(t)-\sum_{j:j\notin G_a} p_j^{k-1}(t)-\sum_{j:j\in G_a} p_j^k(t)$$

Decision variables which don't belong to the subproblem are fixed

- Area-wise subproblems
  - Decouple subproblems w.r.t area-level constraints
  - If more subproblems are desired, then areas are further divided

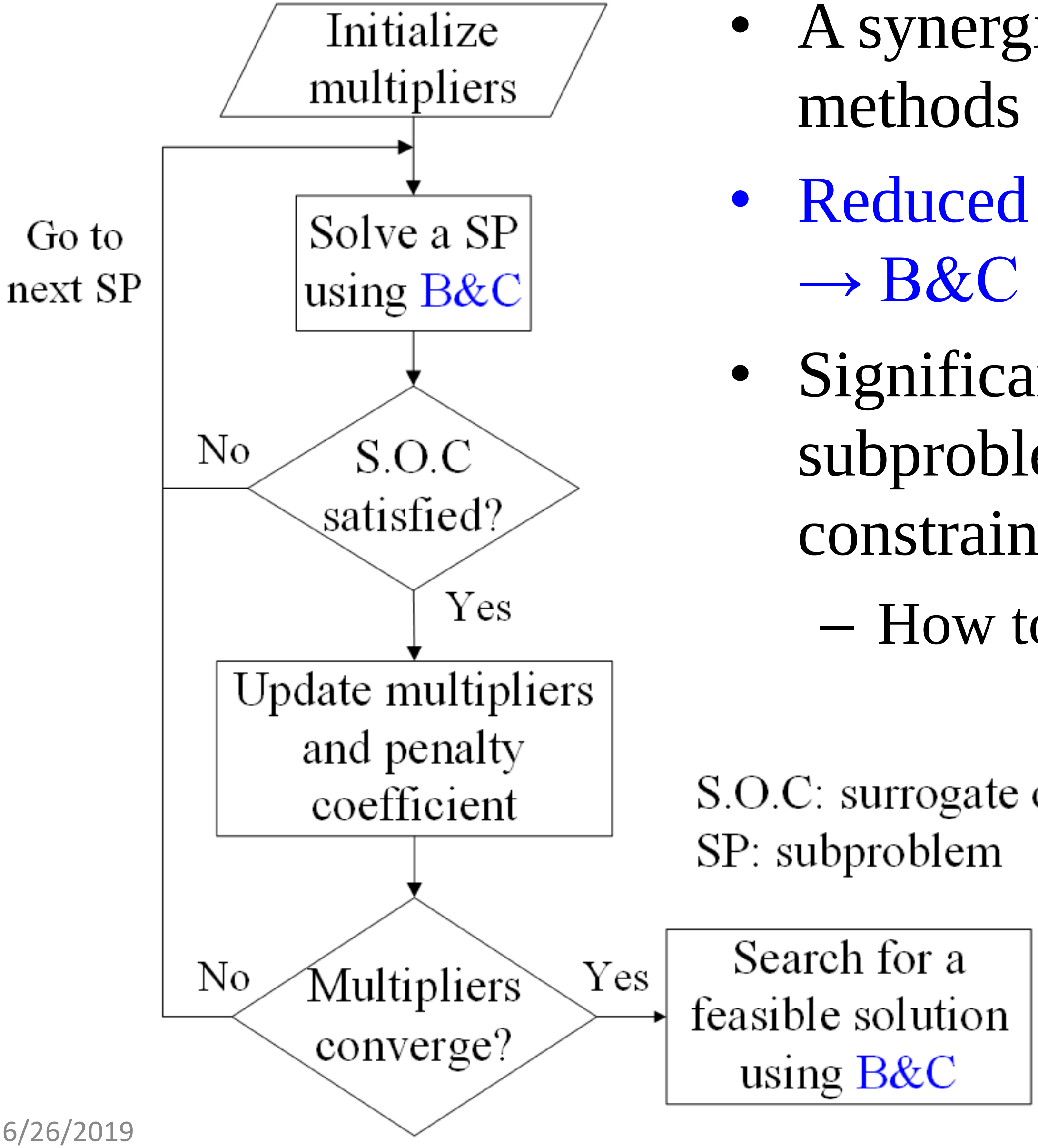
Flow chart



- A synergistic integration of two methods
- Reduced complexity of subproblems → B&C solves subproblems quickly

S.O.C: surrogate optimality condition
SP: subproblem

Flow chart



- A synergistic integration of two methods
- Reduced complexity of subproblems → B&C solves subproblems quickly
- Significant overhead in building subproblem models due to numerous constraints
  - How to decrease overhead?

S.O.C: surrogate optimality condition
SP: subproblem

Computational efficiency improvement

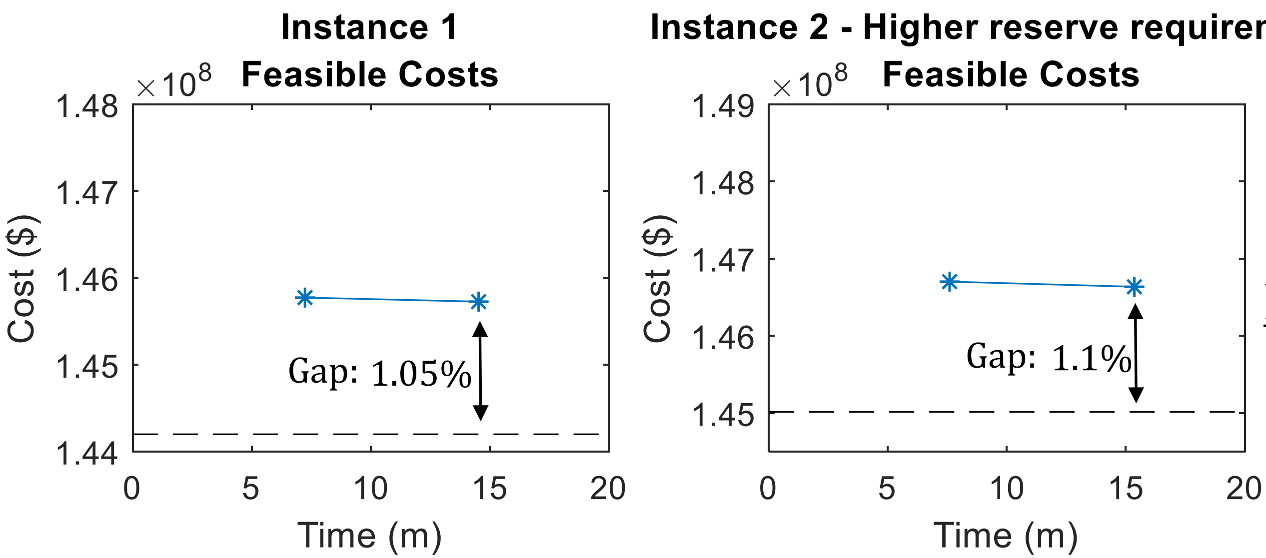
- Most parts of the subproblem model doesn't change at each iteration (e.g., constraint matrix)
- Change the modeling language from OPL to MATLAB
  - Vectorization of loops for building constraints → reduce time
  - To improve efficiency, only the components which change are built at each iteration (e.g., multiplier terms in the objective function)
  - Furthermore, subproblem models are built simultaneously by utilizing parallel processors

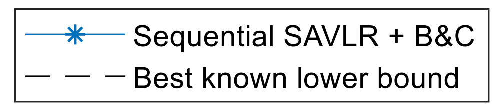
Numerical testing – Sequential SAVLR +B&C

- Polish system (327 units, 2383 buses, 2896 lines, 6 areas)
- The problem is decomposed into 20 subproblems (subareas)
- 2 instances of the problem are solved

Numerical testing – Sequential SAVLR +B&C

- Polish system (327 units, 2383 buses, 2896 lines, 6 areas)
- The problem is decomposed into 20 subproblems (subareas)
- 2 instances of the problem are solved





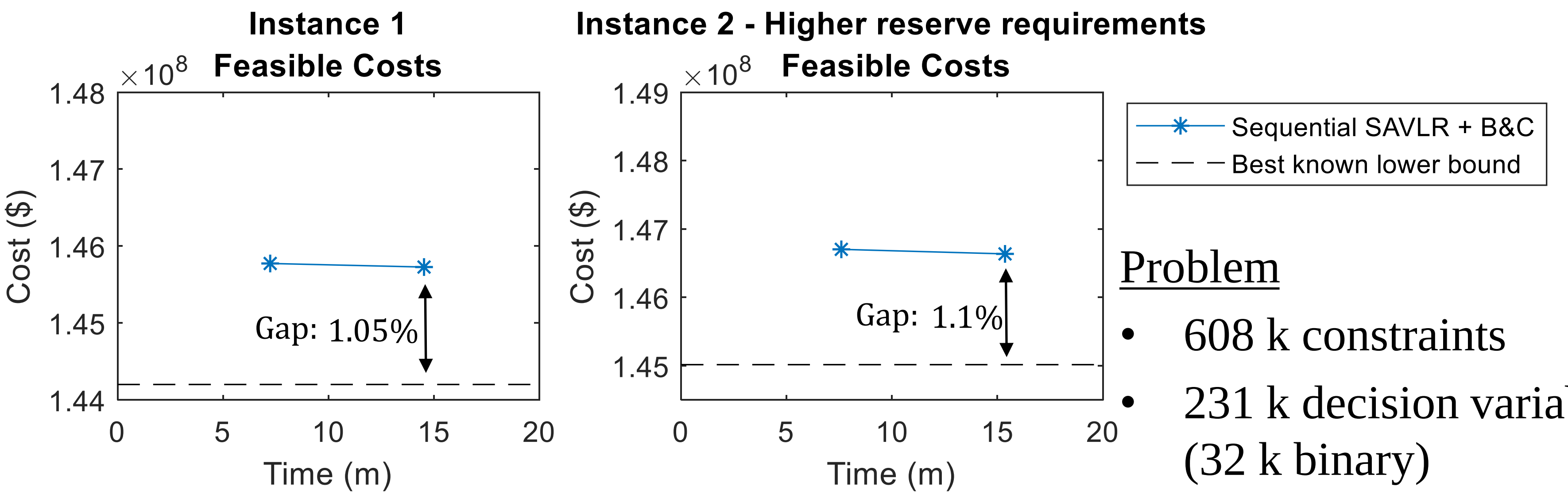
<u>Problem</u>

- 608 k constraints
- 231 k decision variables (32 k binary)

- B&C doesn't find a quality solution within 1 hour
  - 15-min interval → lower ramping capability → higher complexity
- SAVLR+B&C finds near-optimal solutions within 10 minutes

Numerical testing – Sequential SAVLR +B&C

- Polish system (327 units, 2383 buses, 2896 lines, 6 areas)
- The problem is decomposed into 20 subproblems (subareas)
- 2 instances of the problem are solved

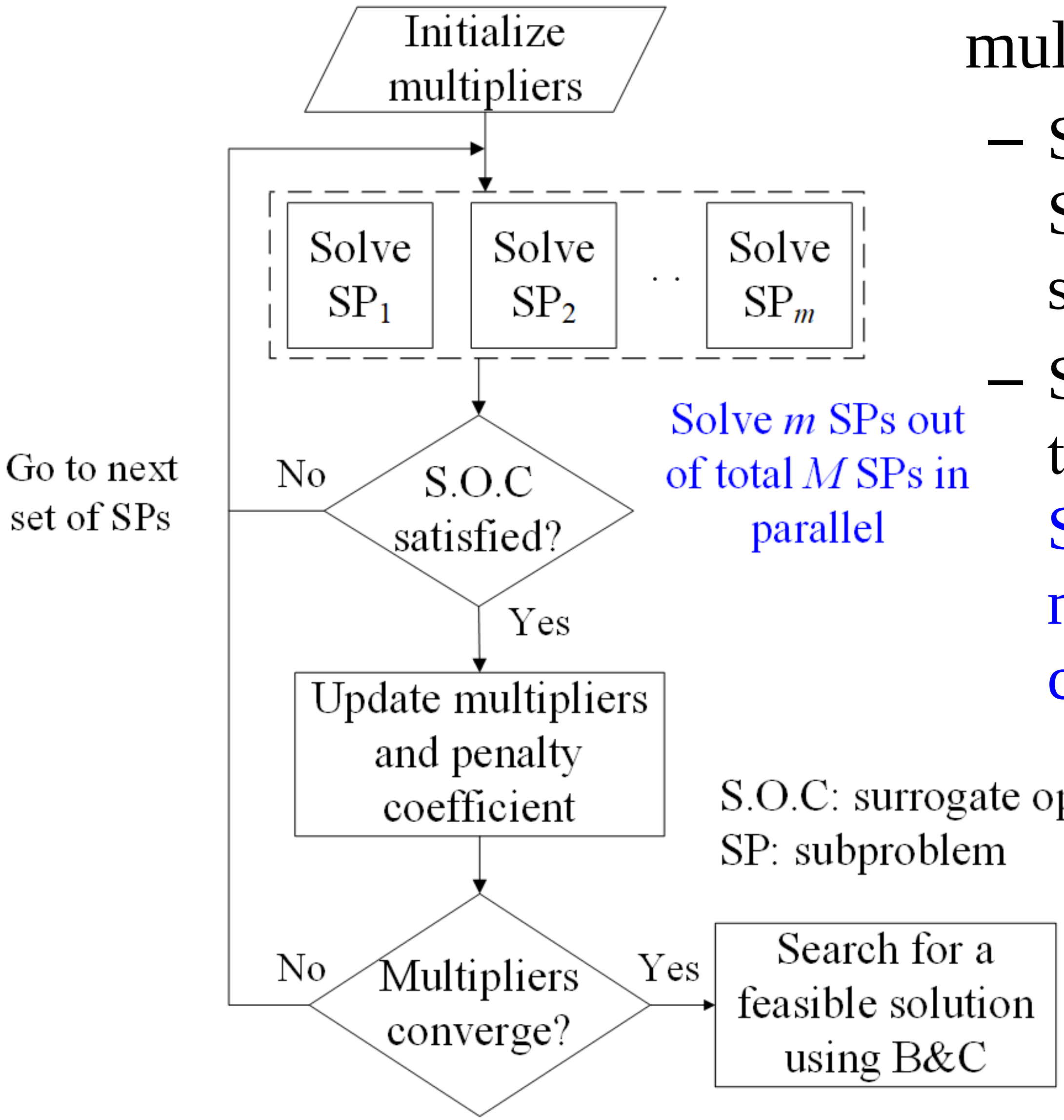


<u>Problem</u>

- 608 k constraints
- 231 k decision variables (32 k binary)

- B&C doesn't find a quality solution within 1 hour
  - 15-min interval → lower ramping capability → higher complexity
- SAVLR+B&C finds near-optimal solutions within 10 minutes
- Can performance be further sped up? Multiple processors?

Parallel SAVLR + B&C

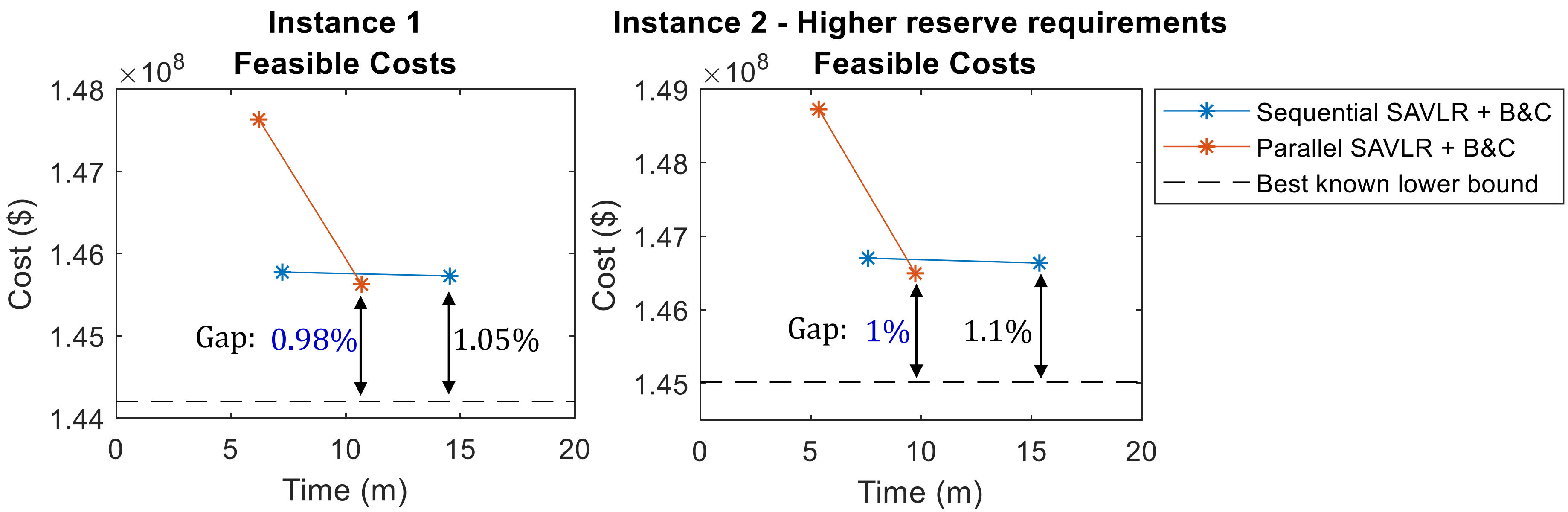


- How can SAVLR utilize multiple processors efficiently?
  - Solve all subproblems → Similar to traditional LR → slow convergence
  - Solve a few subproblems at a time → Preserve the spirit of SAVLR while utilizing multiple processors → faster convergence

S.O.C: surrogate optimality condition
SP: subproblem

Numerical testing – Parallel SAVLR + B&C

- 4 subproblems are solved in parallel



- Parallel SAVLR finds near-optimal solutions quicker than sequential SAVLR
  - Smooth directions for updating multipliers are obtained quickly by solving a few of subproblems simultaneously

Concluding remarks

- SAVLR is a vast improvement over traditional LR
  - Exploit separability where B&C cannot → reduce complexity
  - Surrogate subgradient directions + novel stepsizing rule + absolute value penalties + selective relaxation of constraints → faster and guaranteed convergence
  - Further speed up by efficient parallelization

Concluding remarks

- SAVLR is a vast improvement over traditional LR
  - Exploit separability where B&C cannot → reduce complexity
  - Surrogate subgradient directions + novel stepsizing rule + absolute value penalties + selective relaxation of constraints → faster and guaranteed convergence
  - Further speed up by efficient parallelization
- Exciting results
  - SAVLR+B&C finds near-optimal solutions within 10 minutes while B&C cannot
  - Parallel SAVLR finds near-optimal solutions even quicker

Concluding remarks

- SAVLR is a vast improvement over traditional LR
  - Exploit separability where B&C cannot → reduce complexity
  - Surrogate subgradient directions + novel stepsizing rule + absolute value penalties + selective relaxation of constraints → faster and guaranteed convergence
  - Further speed up by efficient parallelization
- Exciting results
  - SAVLR+B&C finds near-optimal solutions within 10 minutes while B&C cannot
  - Parallel SAVLR finds near-optimal solutions even quicker
- The algorithm is generic and can be used to solve other complex problems in power systems and beyond

Concluding remarks

- SAVLR is a vast improvement over traditional LR
  - Exploit separability where B&C cannot → reduce complexity
  - Surrogate subgradient directions + novel stepsizing rule + absolute value penalties + selective relaxation of constraints → faster and guaranteed convergence
  - Further speed up by efficient parallelization
- Exciting results
  - SAVLR+B&C finds near-optimal solutions within 10 minutes while B&C cannot
  - Parallel SAVLR finds near-optimal solutions even quicker
- The algorithm is generic and can be used to solve other complex problems in power systems and beyond

Thank you!