An optimization heuristic for residential load management

Claudio Giovanni Mattera

Politecnico di Milano (prof. E. Amaldi) - Lund University (prof K. Åström)

14th December 2012

Outline

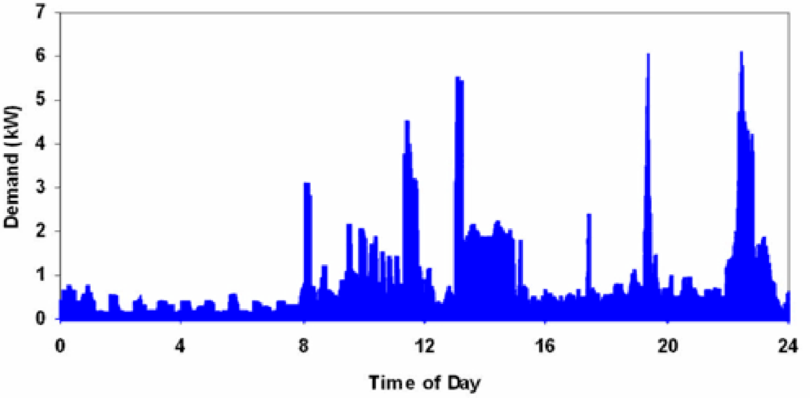
Energy demand curve

- Residential users energy consumption is unsteady;
- most of the time low consumption, but short high peaks;
- facilities and grid are oversized;
- renewable sources are not suitable for short peaks;
- with *smart grids* users aware of changes in the grid's status;
- energy is expensive in peak hours and cheap in off-peak hours;
- hard to manually respond.

Energy demand curve

- Residential users energy consumption is unsteady;
- most of the time low consumption, but short high peaks;
- facilities and grid are oversized;
- renewable sources are not suitable for short peaks;
- with *smart grids* users aware of changes in the grid's status;
- energy is expensive in peak hours and cheap in off-peak hours;
- hard to manually respond.

Energy demand curve - example



Energy demand curve

- Residential users energy consumption is unsteady;
- most of the time low consumption, but short high peaks;
- facilities and grid are oversized;
- renewable sources are not suitable for short peaks;
- with *smart grids* users aware of changes in the grid's status;
- energy is expensive in peak hours and cheap in off-peak hours;
- hard to manually respond.

Energy demand curve

- Residential users energy consumption is unsteady;
- most of the time low consumption, but short high peaks;
- facilities and grid are oversized;
- renewable sources are not suitable for short peaks;
- with *smart grids* users aware of changes in the grid's status;
- energy is expensive in peak hours and cheap in off-peak hours;
- hard to manually respond.

Energy demand curve

- Residential users energy consumption is unsteady;
- most of the time low consumption, but short high peaks;
- facilities and grid are oversized;
- renewable sources are not suitable for short peaks;
- with *smart grids* users aware of changes in the grid's status;
- energy is expensive in peak hours and cheap in off-peak hours;
- hard to manually respond.

Energy demand curve

- Residential users energy consumption is unsteady;
- most of the time low consumption, but short high peaks;
- facilities and grid are oversized;
- renewable sources are not suitable for short peaks;
- with *smart grids* users aware of changes in the grid's status;
- energy is expensive in peak hours and cheap in off-peak hours;
- hard to manually respond.

Energy demand curve

- Residential users energy consumption is unsteady;
- most of the time low consumption, but short high peaks;
- facilities and grid are oversized;
- renewable sources are not suitable for short peaks;
- with *smart grids* users aware of changes in the grid's status;
- energy is expensive in peak hours and cheap in off-peak hours;
- hard to manually respond.

Related work

- Livengood and Larson (2009) introduced the *energy box*, a device to automatically control appliances in a house;
- Mohsenian-Rad and Leon-Garcia (2010) proposed an algorithm to forecast energy's price variations in a dynamic pricing policy;
- Kowahl and Kuh (2010) adapted the framework proposed by Livengood and Larson (2009) to more realistic scenarios, using a reinforcement learning approach;
- Kishore and Snyder (2010) showed that single houses cost minimization results only in peaks shift, and proposed a multiple house approach;
- Agnetis et al. (2011) proposed a system where the energy retailer signals the users, which receive rewards if adapt their consumption;
- Barbato et al. (2011a,b) proposed a MILP model to minimize the total cost for cooperative and non-cooperative residential users; our thesis is based on an extension of this model.

Residential energy management problem

- The day is discretized in 96 time slots of 15 minutes;
- each appliance has an *starting window*;
- each appliance consumes different amounts of energy in different phases (load profile);
- a house can buy limited amount of energy in a time slot;
- photovoltaic panels produce known amount of energy;
- energy can be stored in batteries for later usage.

Given the data above, decide the starting time slot for each appliance, in order to minimize an objective function.
The purpose of the thesis was to develop heuristics to generate high quality solutions in reasonable time.

Residential energy management problem

- The day is discretized in 96 time slots of 15 minutes;
- each appliance has an *starting window*;
- each appliance consumes different amounts of energy in different phases (load profile);
- a house can buy limited amount of energy in a time slot;
- photovoltaic panels produce known amount of energy;
- energy can be stored in batteries for later usage.

Given the data above, decide the starting time slot for each appliance, in order to minimize an objective function.
The purpose of the thesis was to develop heuristics to generate high quality solutions in reasonable time.

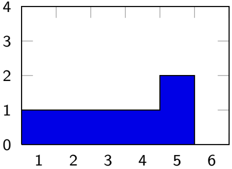
Residential energy management problem

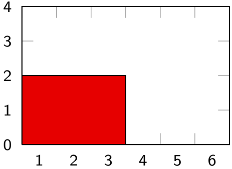
- The day is discretized in 96 time slots of 15 minutes;
- each appliance has an *starting window*;
- each appliance consumes different amounts of energy in different phases (load profile);
- a house can buy limited amount of energy in a time slot;
- photovoltaic panels produce known amount of energy;
- energy can be stored in batteries for later usage.

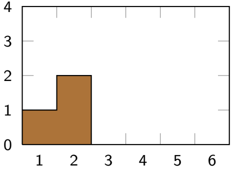
Given the data above, decide the starting time slot for each appliance, in order to minimize an objective function.
The purpose of the thesis was to develop heuristics to generate high quality solutions in reasonable time.

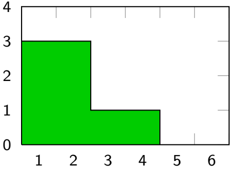
Load profiles

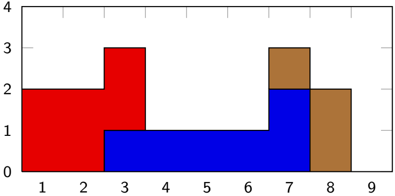
Load profiles of four appliances , , and , and resulting demand curve for the first three.











Residential energy management problem

- The day is discretized in 96 time slots of 15 minutes;
- each appliance has an *starting window*;
- each appliance consumes different amounts of energy in different phases (load profile);
- a house can buy limited amount of energy in a time slot;
- photovoltaic panels produce known amount of energy;
- energy can be stored in batteries for later usage.

Given the data above, decide the starting time slot for each appliance, in order to minimize an objective function.
The purpose of the thesis was to develop heuristics to generate high quality solutions in reasonable time.

Residential energy management problem

- The day is discretized in 96 time slots of 15 minutes;
- each appliance has an *starting window*;
- each appliance consumes different amounts of energy in different phases (load profile);
- a house can buy limited amount of energy in a time slot;
- photovoltaic panels produce known amount of energy;
- energy can be stored in batteries for later usage.

Given the data above, decide the starting time slot for each appliance, in order to minimize an objective function.
The purpose of the thesis was to develop heuristics to generate high quality solutions in reasonable time.

Residential energy management problem

- The day is discretized in 96 time slots of 15 minutes;
- each appliance has an *starting window*;
- each appliance consumes different amounts of energy in different phases (load profile);
- a house can buy limited amount of energy in a time slot;
- photovoltaic panels produce known amount of energy;
- energy can be stored in batteries for later usage.

Given the data above, decide the starting time slot for each appliance, in order to minimize an objective function.
The purpose of the thesis was to develop heuristics to generate high quality solutions in reasonable time.

Residential energy management problem

- The day is discretized in 96 time slots of 15 minutes;
- each appliance has an *starting window*;
- each appliance consumes different amounts of energy in different phases (load profile);
- a house can buy limited amount of energy in a time slot;
- photovoltaic panels produce known amount of energy;
- energy can be stored in batteries for later usage.

Given the data above, decide the starting time slot for each appliance, in order to minimize an objective function.

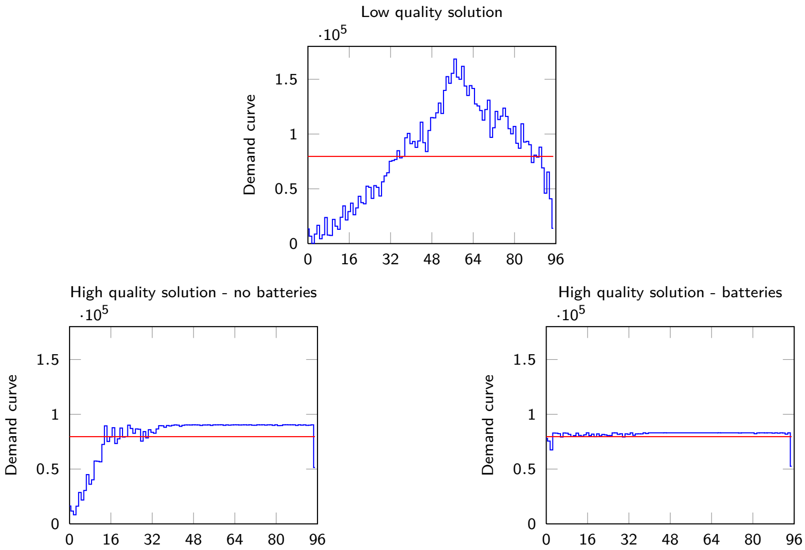
The purpose of the thesis was to develop heuristics to generate high quality solutions in reasonable time.

Residential energy management problem

- The day is discretized in 96 time slots of 15 minutes;
- each appliance has an *starting window*;
- each appliance consumes different amounts of energy in different phases (load profile);
- a house can buy limited amount of energy in a time slot;
- photovoltaic panels produce known amount of energy;
- energy can be stored in batteries for later usage.

Given the data above, decide the starting time slot for each appliance, in order to minimize an objective function.
The purpose of the thesis was to develop heuristics to generate high quality solutions in reasonable time.

Example with real instances



Objective functions

To measure the quality of solutions we used different objective functions.

- The aim is to minimize the maximal peak of demand curve;
- maximal peak suffers from bottleneck;
- area of demand curve is constant, the *ideal* curve is completely flat;
- we minimize a distance between the current curve and the ideal one;
- p-norm of difference effectively distinguish between solutions.

Objective functions

To measure the quality of solutions we used different objective functions.

- The aim is to minimize the maximal peak of demand curve;
- maximal peak suffers from bottleneck;
- area of demand curve is constant, the *ideal* curve is completely flat;
- we minimize a distance between the current curve and the ideal one;
- *p*-norm of difference effectively distinguish between solutions.

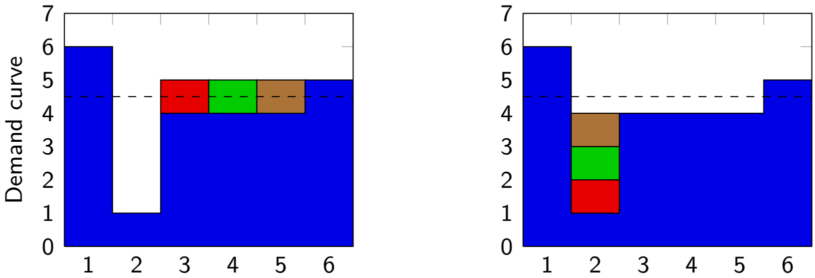
Objective functions

To measure the quality of solutions we used different objective functions.

- The aim is to minimize the maximal peak of demand curve;
- maximal peak suffers from bottleneck;
- area of demand curve is constant, the *ideal* curve is completely flat;
- we minimize a distance between the current curve and the ideal one;
- p-norm of difference effectively distinguish between solutions.

Objective functions - Maximal peak

The two demand curves have the same maximal peak, but the second one is much more regular.



Objective functions

To measure the quality of solutions we used different objective functions.

- The aim is to minimize the maximal peak of demand curve;
- maximal peak suffers from bottleneck;
- area of demand curve is constant, the *ideal* curve is completely flat;
- we minimize a distance between the current curve and the ideal one;
- p-norm of difference effectively distinguish between solutions.

Objective functions

To measure the quality of solutions we used different objective functions.

- The aim is to minimize the maximal peak of demand curve;
- maximal peak suffers from bottleneck;
- area of demand curve is constant, the *ideal* curve is completely flat;
- we minimize a distance between the current curve and the ideal one;
- p-norm of difference effectively distinguish between solutions.

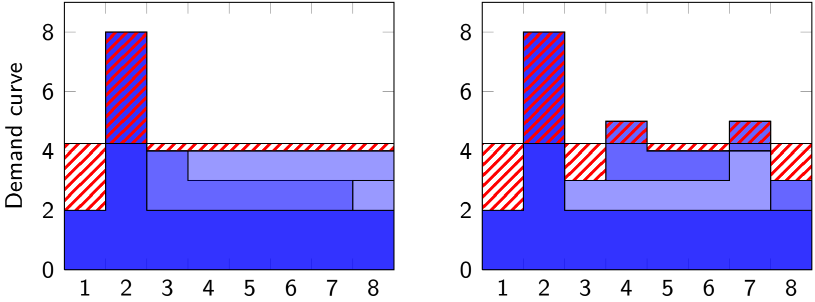
Objective functions

To measure the quality of solutions we used different objective functions.

- The aim is to minimize the maximal peak of demand curve;
- maximal peak suffers from bottleneck;
- area of demand curve is constant, the *ideal* curve is completely flat;
- we minimize a distance between the current curve and the ideal one;
- p-norm of difference effectively distinguish between solutions.

Objective functions - Difference's p-norm

The two demand curves have the same maximal peak and maximal difference from ideal. Area of difference can distinguish between them.



Greedy Randomized Adaptive Search Procedure (GRASP)

GRASP (Feo and Resende, 1995) is a fast approach to generate an initial feasible solution:

- for each appliance:
  - consider all the starting slots within the starting window;
  - for each starting slot compute the resulting demand curve;
  - discard the lowest quality candidates;
  - pick a random candidate among the remaining;
- repeat many times and return the best solution generated.

Greedy Randomized Adaptive Search Procedure (GRASP)

GRASP (Feo and Resende, 1995) is a fast approach to generate an initial feasible solution:

- for each appliance:
  - consider all the starting slots within the starting window;
  - for each starting slot compute the resulting demand curve;
  - discard the lowest quality *candidates*;
  - pick a random candidate among the remaining.
- repeat many times and return the best solution generated.

Greedy Randomized Adaptive Search Procedure (GRASP)

GRASP (Feo and Resende, 1995) is a fast approach to generate an initial feasible solution:

- for each appliance:
  - consider all the starting slots within the starting window;
  - for each starting slot compute the resulting demand curve;
  - discard the lowest quality *candidates*;
  - pick a random candidate among the remaining.
- repeat many times and return the best solution generated.

Greedy Randomized Adaptive Search Procedure (GRASP)

GRASP (Feo and Resende, 1995) is a fast approach to generate an initial feasible solution:

- for each appliance:
  - consider all the starting slots within the starting window;
  - for each starting slot compute the resulting demand curve;
  - discard the lowest quality *candidates*;
  - pick a random candidate among the remaining.
- repeat many times and return the best solution generated.

Greedy Randomized Adaptive Search Procedure (GRASP)

GRASP (Feo and Resende, 1995) is a fast approach to generate an initial feasible solution:

- for each appliance:
  - consider all the starting slots within the starting window;
  - for each starting slot compute the resulting demand curve;
  - discard the lowest quality *candidates*;
  - pick a random candidate among the remaining.
- repeat many times and return the best solution generated.

Greedy Randomized Adaptive Search Procedure (GRASP)

GRASP (Feo and Resende, 1995) is a fast approach to generate an initial feasible solution:

- for each appliance:
  - consider all the starting slots within the starting window;
  - for each starting slot compute the resulting demand curve;
  - discard the lowest quality *candidates*;
  - pick a random candidate among the remaining.
- repeat many times and return the best solution generated.

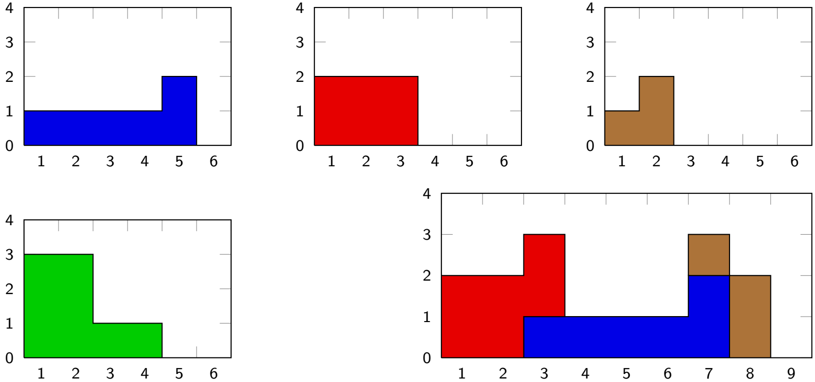
Greedy Randomized Adaptive Search Procedure (GRASP)

GRASP (Feo and Resende, 1995) is a fast approach to generate an initial feasible solution:

- for each appliance:
  - consider all the starting slots within the starting window;
  - for each starting slot compute the resulting demand curve;
  - discard the lowest quality *candidates*;
  - pick a random candidate among the remaining.
- repeat many times and return the best solution generated.

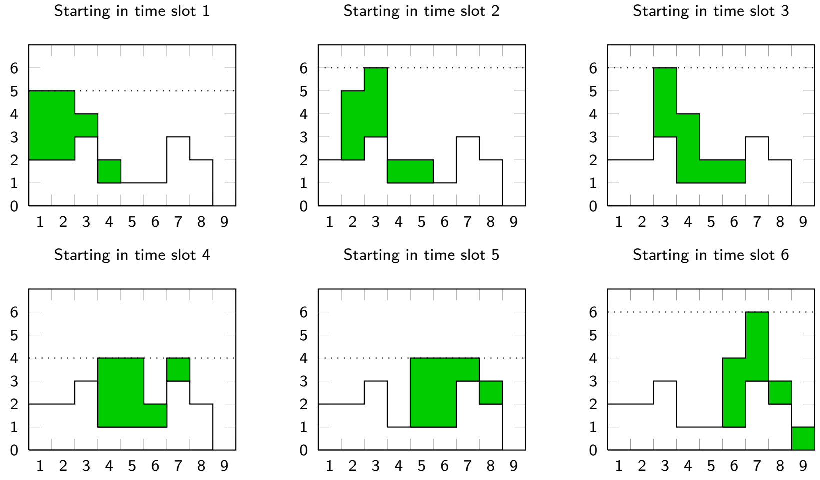
Load profiles

Load profiles of four appliances , , and , and resulting demand curve for the first three.



Example of GRASP iteration

Demand curves for different starting time of activity.



Tabu Search

Tabu Search (Glover and Laguna, 1998) is a local search method less vulnerable to local minima.

- Start from an initial solution and improve it iteratively with its best neighbour;
- neighbourhood is generated by applying moves to the current solution;
- recent solutions (*Tabu moves*) are forbidden, helping to escape from local minima;
- detect if the solution has not improved in recent iterations: stop or diversification;
- strategic oscillation between feasible and infeasible regions;
- extensible framework.

Tabu Search

Tabu Search (Glover and Laguna, 1998) is a local search method less vulnerable to local minima.

- Start from an initial solution and improve it iteratively with its best neighbour;
- neighbourhood is generated by applying moves to the current solution;
- recent solutions (*Tabu moves*) are forbidden, helping to escape from local minima;
- detect if the solution has not improved in recent iterations: stop or diversification;
- strategic oscillation between feasible and infeasible regions;
- extensible framework.

Tabu Search

Tabu Search (Glover and Laguna, 1998) is a local search method less vulnerable to local minima.

- Start from an initial solution and improve it iteratively with its best neighbour;
- neighbourhood is generated by applying moves to the current solution;
- recent solutions (*Tabu moves*) are forbidden, helping to escape from local minima;
- detect if the solution has not improved in recent iterations: stop or diversification;
- strategic oscillation between feasible and infeasible regions;
- extensible framework.

Tabu Search

Tabu Search (Glover and Laguna, 1998) is a local search method less vulnerable to local minima.

- Start from an initial solution and improve it iteratively with its best neighbour;
- neighbourhood is generated by applying moves to the current solution;
- recent solutions (*Tabu moves*) are forbidden, helping to escape from local minima;
- detect if the solution has not improved in recent iterations: stop or diversification;
- strategic oscillation between feasible and infeasible regions;
- extensible framework.

Tabu Search

Tabu Search (Glover and Laguna, 1998) is a local search method less vulnerable to local minima.

- Start from an initial solution and improve it iteratively with its best neighbour;
- neighbourhood is generated by applying moves to the current solution;
- recent solutions (*Tabu moves*) are forbidden, helping to escape from local minima;
- detect if the solution has not improved in recent iterations: stop or diversification;
- strategic oscillation between feasible and infeasible regions;
- extensible framework.

Tabu Search

Tabu Search (Glover and Laguna, 1998) is a local search method less vulnerable to local minima.

- Start from an initial solution and improve it iteratively with its best neighbour;
- neighbourhood is generated by applying moves to the current solution;
- recent solutions (*Tabu moves*) are forbidden, helping to escape from local minima;
- detect if the solution has not improved in recent iterations: stop or diversification;
- strategic oscillation between feasible and infeasible regions;
- extensible framework.

Tabu Search

Tabu Search (Glover and Laguna, 1998) is a local search method less vulnerable to local minima.

- Start from an initial solution and improve it iteratively with its best neighbour;
- neighbourhood is generated by applying moves to the current solution;
- recent solutions (*Tabu moves*) are forbidden, helping to escape from local minima;
- detect if the solution has not improved in recent iterations: stop or diversification;
- strategic oscillation between feasible and infeasible regions;
- extensible framework.

Different moves to explore the neighbourhood

Shift
: Changing an appliance's starting time slot to an arbitrary (feasible) value;

Swap
: Exchanging two appliances starting time slots;

Battery
: Buying in advance the energy used in a time slot, storing it in a battery;

MILP
: Letting a MILP solver to find an improving scheduling, fixing some appliances;

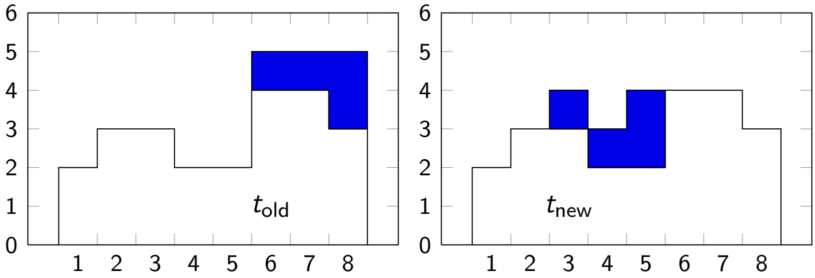
MILP-batteries
: Letting a MILP solver to find the best batteries usage for the current scheduling;

MILP-zeros
: Discarding $N\%$ of *unused* starting slots and letting a MILP solver to find an improving solution;

Large
: Rescheduling few appliances with GRASP;

Mixed
: Picking at runtime the best move.

Shifting an activity



Different moves to explore the neighbourhood

Shift Changing an appliance's starting time slot to an arbitrary (feasible) value;

Swap Exchanging two appliances starting time slots;

Battery Buying in advance the energy used in a time slot, storing it in a battery;

MILP Letting a MILP solver to find an improving scheduling, fixing some appliances;

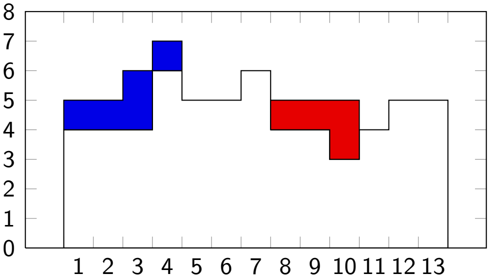
MILP-batteries Letting a MILP solver to find the best batteries usage for the current scheduling;

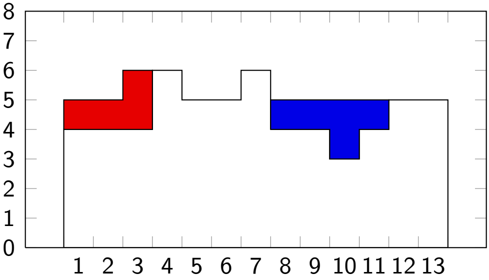
MILP-zeros Discarding $N\%$ of *unused* starting slots and letting a MILP solver to find an improving solution;

Large Rescheduling few appliances with GRASP;

Mixed Picking at runtime the best move.

Swapping two activities



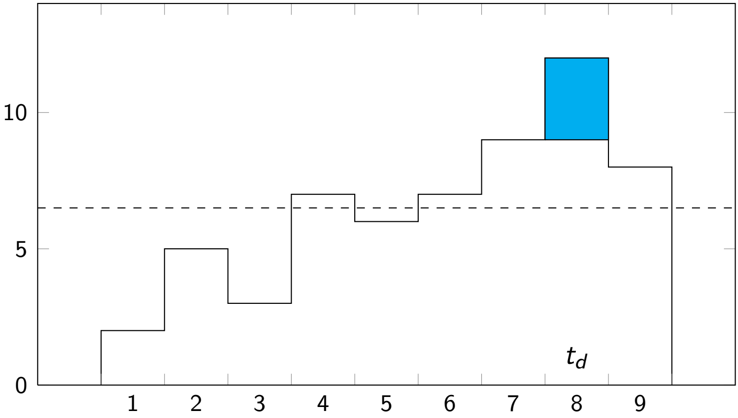


Different moves to explore the neighbourhood

- **Shift** Changing an appliance's starting time slot to an arbitrary (feasible) value;
- **Swap** Exchanging two appliances starting time slots;
- **Battery** Buying in advance the energy used in a time slot, storing it in a battery;
- **MILP** Letting a MILP solver to find an improving scheduling, fixing some appliances;
- **MILP-batteries** Letting a MILP solver to find the best batteries usage for the current scheduling;
- **MILP-zeros** Discarding N% of *unused* starting slots and letting a MILP solver to find an improving solution;
- **Large** Rescheduling few appliances with GRASP;
- **Mixed** Picking at runtime the best move.

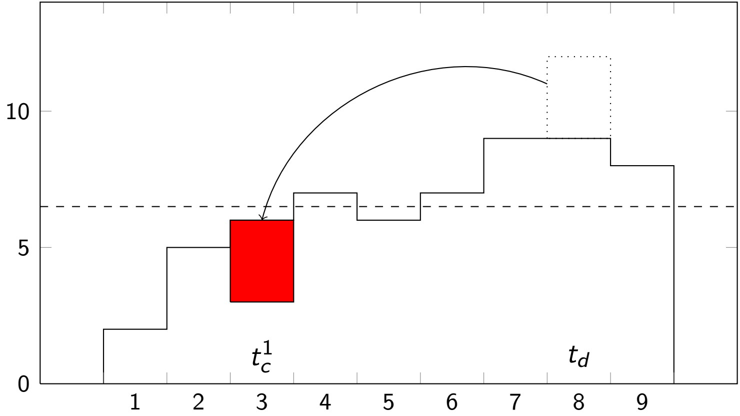
Battery move example

In time slot t_d a house with battery buys an amount of energy ▭.



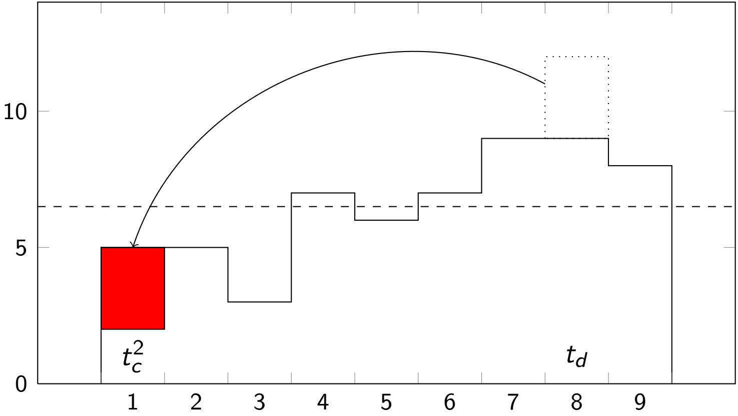
Battery move example (2)

We can anticipate such load in time slot t_c^1.



Battery move example (3)

Or in time slot t_c^2.



Different moves to explore the neighbourhood

- **Shift** Changing an appliance's starting time slot to an arbitrary (feasible) value;
- **Swap** Exchanging two appliances starting time slots;
- **Battery** Buying in advance the energy used in a time slot, storing it in a battery;
- **MILP** Letting a MILP solver to find an improving scheduling, fixing some appliances;
- **MILP-batteries** Letting a MILP solver to find the best batteries usage for the current scheduling;
- **MILP-zeros** Discarding N% of *unused* starting slots and letting a MILP solver to find an improving solution;
- **Large** Rescheduling few appliances with GRASP;
- **Mixed** Picking at runtime the best move.

Different moves to explore the neighbourhood

- **Shift** Changing an appliance's starting time slot to an arbitrary (feasible) value;
- **Swap** Exchanging two appliances starting time slots;
- **Battery** Buying in advance the energy used in a time slot, storing it in a battery;
- **MILP** Letting a MILP solver to find an improving scheduling, fixing some appliances;
- **MILP-batteries** Letting a MILP solver to find the best batteries usage for the current scheduling;
- **MILP-zeros** Discarding $N\%$ of *unused* starting slots and letting a MILP solver to find an improving solution;
- **Large** Rescheduling few appliances with GRASP;
- **Mixed** Picking at runtime the best move.

Different moves to explore the neighbourhood

- **Shift** Changing an appliance's starting time slot to an arbitrary (feasible) value;
- **Swap** Exchanging two appliances starting time slots;
- **Battery** Buying in advance the energy used in a time slot, storing it in a battery;
- **MILP** Letting a MILP solver to find an improving scheduling, fixing some appliances;
- **MILP-batteries** Letting a MILP solver to find the best batteries usage for the current scheduling;
- **MILP-zeros** Discarding $N\%$ of *unused* starting slots and letting a MILP solver to find an improving solution;
- **Large** Rescheduling few appliances with GRASP;
- **Mixed** Picking at runtime the best move.

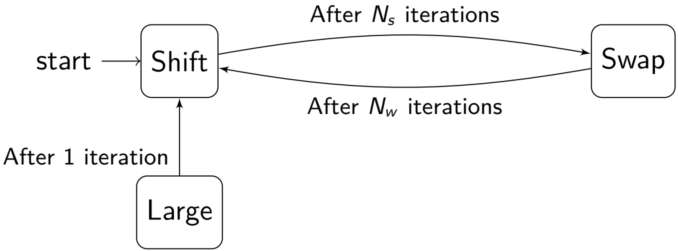
Different moves to explore the neighbourhood

Shift	Changing an appliance's starting time slot to an arbitrary (feasible) value;
Swap	Exchanging two appliances starting time slots;
Battery	Buying in advance the energy used in a time slot, storing it in a battery;
MILP	Letting a MILP solver to find an improving scheduling, fixing some appliances;
MILP-batteries	Letting a MILP solver to find the best batteries usage for the current scheduling;
MILP-zeros	Discarding $N\%$ of *unused* starting slots and letting a MILP solver to find an improving solution;
Large	Rescheduling few appliances with GRASP;
Mixed	Picking at runtime the best move.

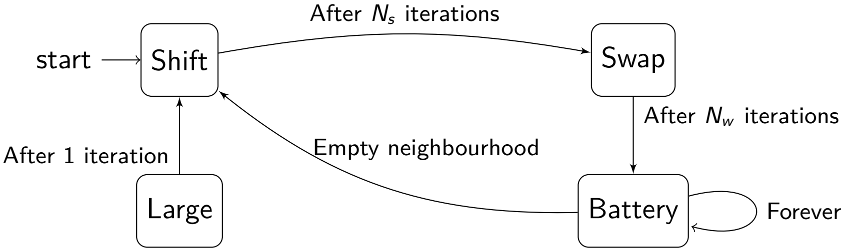
Different moves to explore the neighbourhood

- **Shift** Changing an appliance's starting time slot to an arbitrary (feasible) value;
- **Swap** Exchanging two appliances starting time slots;
- **Battery** Buying in advance the energy used in a time slot, storing it in a battery;
- **MILP** Letting a MILP solver to find an improving scheduling, fixing some appliances;
- **MILP-batteries** Letting a MILP solver to find the best batteries usage for the current scheduling;
- **MILP-zeros** Discarding $N\%$ of *unused* starting slots and letting a MILP solver to find an improving solution;
- **Large** Rescheduling few appliances with GRASP;
- **Mixed** Picking at runtime the best move.

Mixed move - No batteries



Mixed move - Battery moves



Data set

- 180 instances with different number of houses, PV panels and batteries;

Houses	PV panels	Batteries
20	0, 2, 4	0, 2
200	0, 20, 40	0, 20
400	0, 40, 80	0, 40

- instances were generated from a realistic set of parameters obtained in previous work;
- our methods perform heterogeneously on instances with different number of houses/batteries;
- average gap from reference for aggregate.

Data set

- 180 instances with different number of houses, PV panels and batteries;

Houses	PV panels	Batteries
20	0, 2, 4	0, 2
200	0, 20, 40	0, 20
400	0, 40, 80	0, 40

- instances were generated from a realistic set of parameters obtained in previous work;
- our methods perform heterogeneously on instances with different number of houses/batteries;
- average gap from reference for aggregate.

Data set

- 180 instances with different number of houses, PV panels and batteries;

Houses	PV panels	Batteries
20	0, 2, 4	0, 2
200	0, 20, 40	0, 20
400	0, 40, 80	0, 40

- instances were generated from a realistic set of parameters obtained in previous work;
- our methods perform heterogeneously on instances with different number of houses/batteries;
- average gap from reference for aggregate.

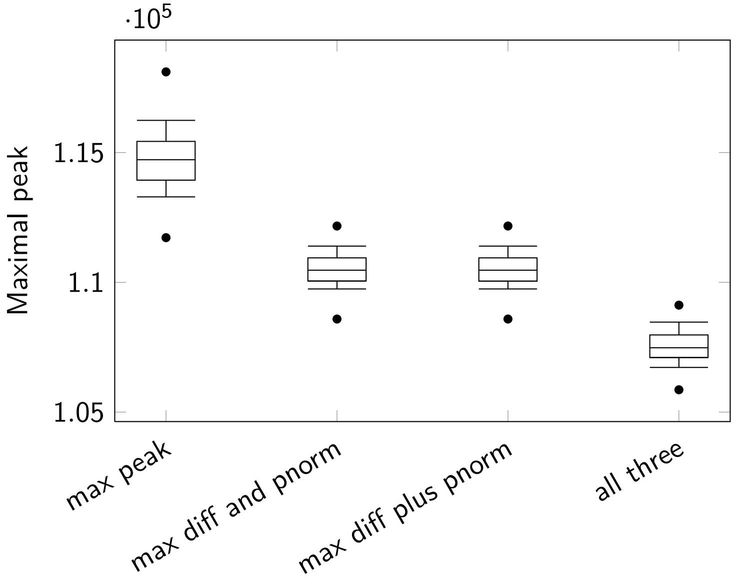
Data set

- 180 instances with different number of houses, PV panels and batteries;

Houses	PV panels	Batteries
20	0, 2, 4	0, 2
200	0, 20, 40	0, 20
400	0, 40, 80	0, 40

- instances were generated from a realistic set of parameters obtained in previous work;
- our methods perform heterogeneously on instances with different number of houses/batteries;
- average gap from reference for aggregate.

Objective functions



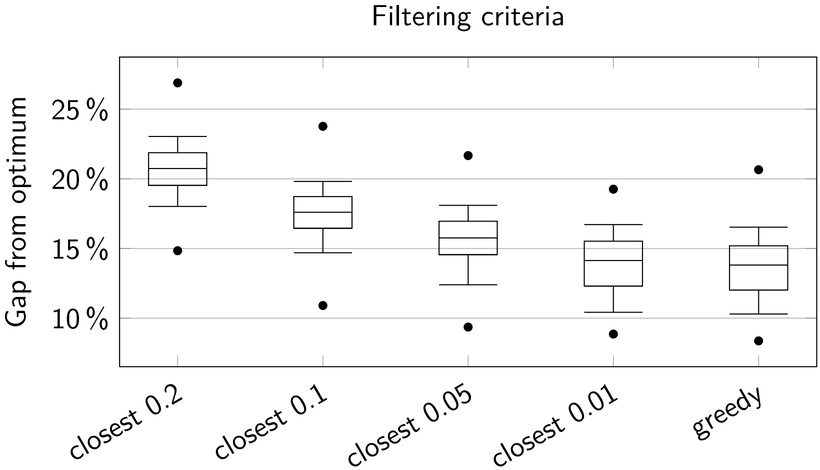
GRASP results

Criteria	20 houses		200 houses		400 houses	
	Gap	Time	Gap	Time	Gap	Time
Closest 0.2	28.15%	2.93	20.66%	59.33	20.85%	213.27
Closest 0.1	25.11%	2.93	17.46%	59.47	17.69%	202.67
Closest 0.05	23.40%	2.9	15.59%	60.2	15.83%	222.2
Closest 0.01	21.70%	2.83	13.89%	63.17	13.77%	209.3
Greedy	21.05%	2.7	13.63%	52.87	13.36%	194.0

Generated 1000 solutions. GRASP generates better solutions when for each appliance keeps only the best starting slots.

GRASP results - continued

Distribution of gap from optimum for 200 houses, 0 batteries instances.



Tabu Search results

- We generated 10 solutions with GRASP and improved the best one;
- computing time includes GRASP time;
- comparison with Partial Linear Relaxation (PLR).

Tabu Search results

- We generated 10 solutions with GRASP and improved the best one;
- computing time includes GRASP time;
- comparison with Partial Linear Relaxation (PLR).
  - Solve a PLR and a reduced MILP, followed by Tabu Search;
  - best results, but not all instances solved.

Tabu Search results

- We generated 10 solutions with GRASP and improved the best one;
- computing time includes GRASP time;
- comparison with Partial Linear Relaxation (PLR).
  - Solve a PLR and a reduced MILP, followed by Tabu Search;
  - best results, but not all instances solved.

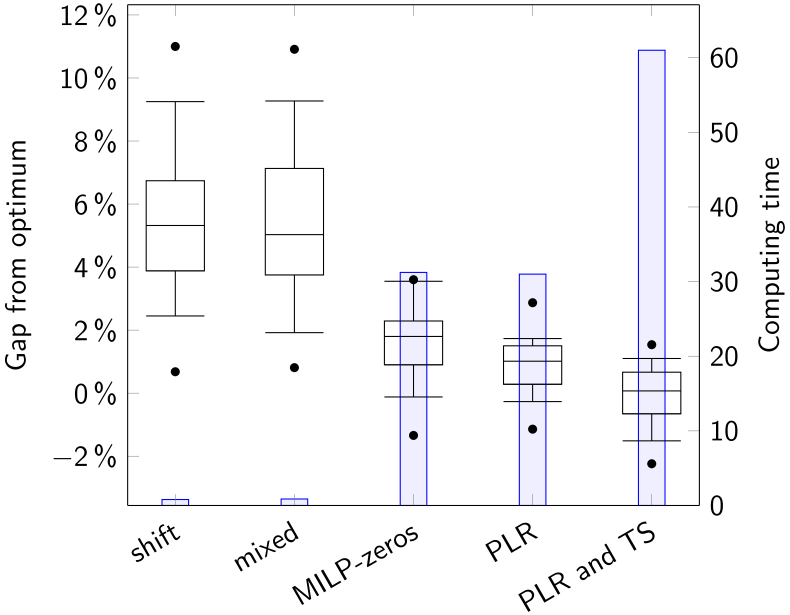
Tabu Search results

- We generated 10 solutions with GRASP and improved the best one;
- computing time includes GRASP time;
- comparison with Partial Linear Relaxation (PLR).
  - Solve a PLR and a reduced MILP, followed by Tabu Search;
  - best results, but not all instances solved.

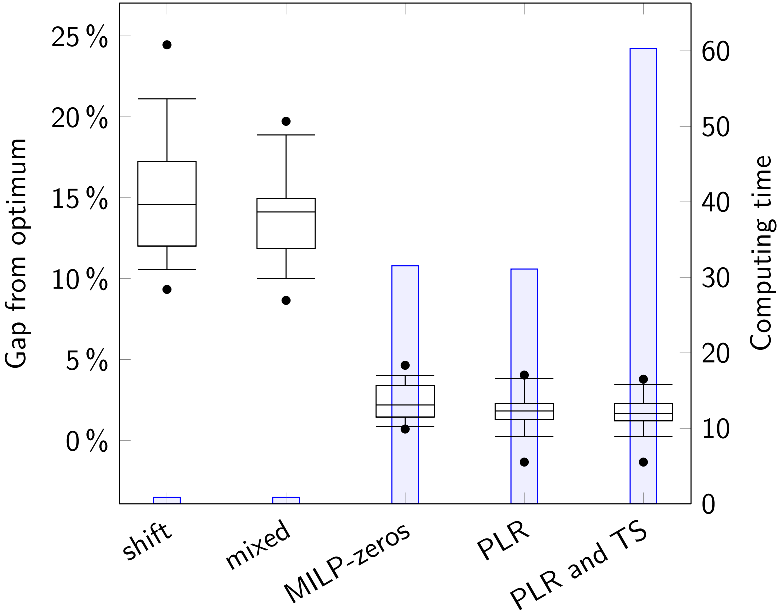
Tabu Search results

- We generated 10 solutions with GRASP and improved the best one;
- computing time includes GRASP time;
- comparison with Partial Linear Relaxation (PLR).
  - Solve a PLR and a reduced MILP, followed by Tabu Search;
  - best results, but not all instances solved.

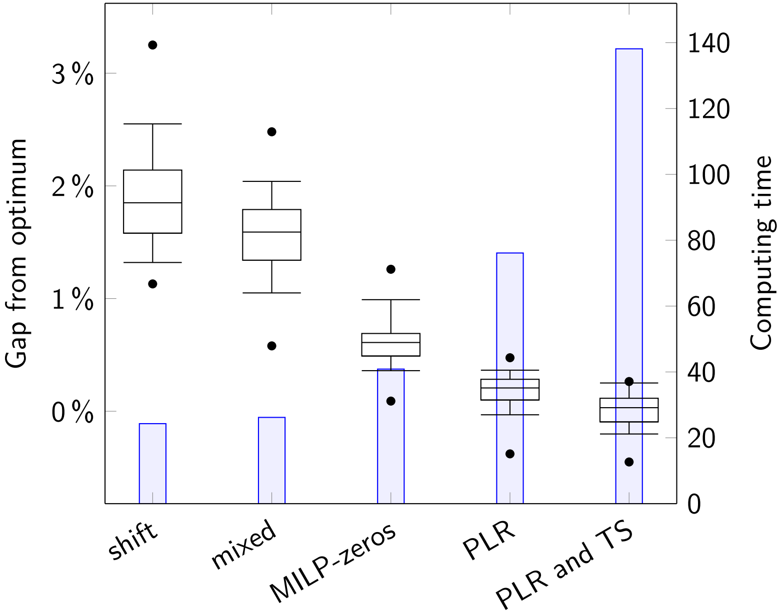
Tabu Search - 20 houses, 0 batteries



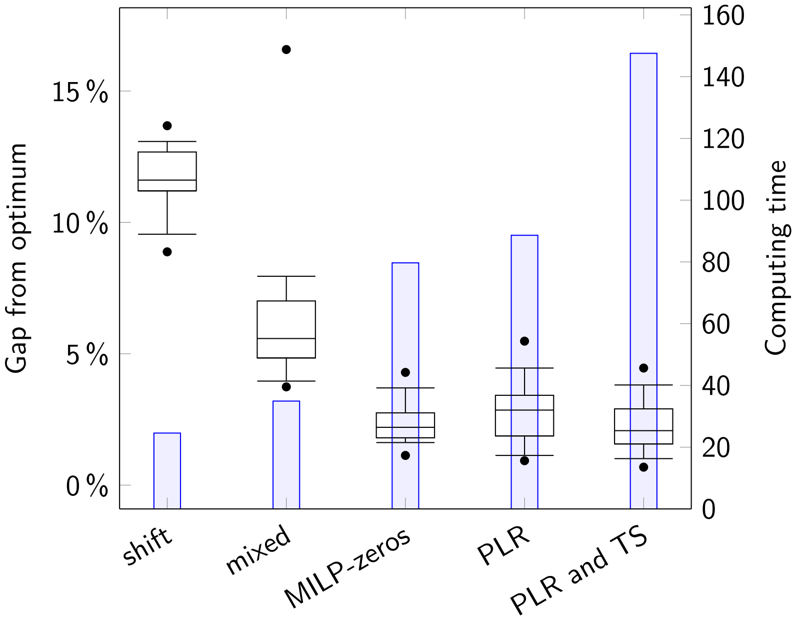
Tabu Search - 20 houses, 2 batteries



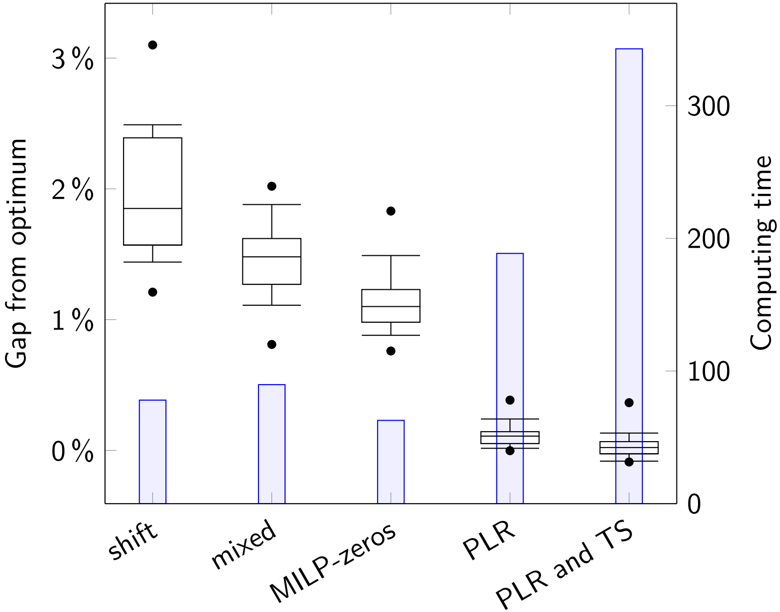
Tabu Search - 200 houses, 0 batteries



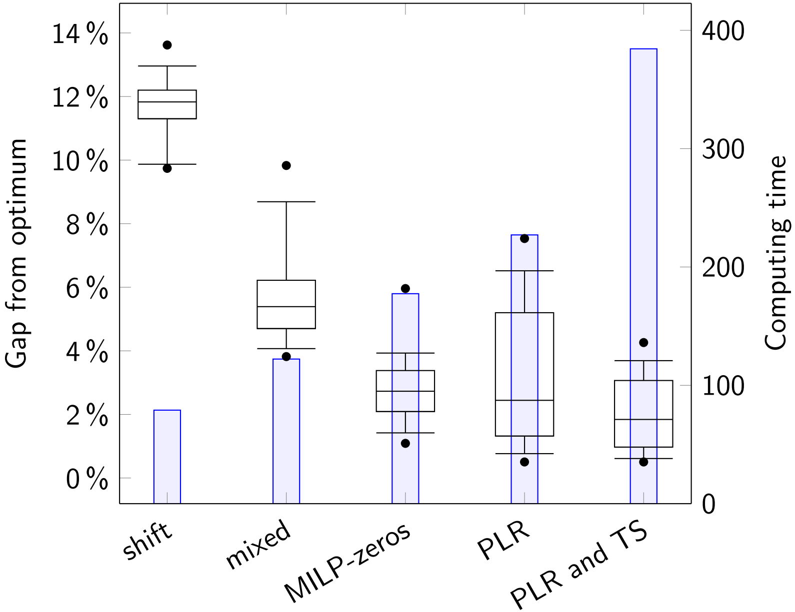
Tabu Search - 200 houses, 20 batteries



Tabu Search - 400 houses, 0 batteries



Tabu Search - 400 houses, 40 batteries



Concluding remarks

- GRASP very fast but produces low quality solutions;
- several moves for Tabu Search;
- trade-off between solution's quality and computing time;
- best results when embedding reduced MILPs in the heuristic.

We achieved producing high quality solutions in short time.
Tabu Search with MILP-zeros move is the best compromise.
For instances without batteries, shift move.
For much larger instances, mixed move.

Concluding remarks

- GRASP very fast but produces low quality solutions;
- several moves for Tabu Search;
- trade-off between solution's quality and computing time;
- best results when embedding reduced MILPs in the heuristic.

We achieved producing high quality solutions in short time.
Tabu Search with MILP-zeros move is the best compromise.
For instances without batteries, shift move.
For much larger instances, mixed move.

Concluding remarks

- GRASP very fast but produces low quality solutions;
- several moves for Tabu Search;
- trade-off between solution's quality and computing time;
- best results when embedding reduced MILPs in the heuristic.

We achieved producing high quality solutions in short time.
Tabu Search with MILP-zeros move is the best compromise.
For instances without batteries, shift move.
For much larger instances, mixed move.

Concluding remarks

- GRASP very fast but produces low quality solutions;
- several moves for Tabu Search;
- trade-off between solution's quality and computing time;
- best results when embedding reduced MILPs in the heuristic.

We achieved producing high quality solutions in short time.
Tabu Search with MILP-zeros move is the best compromise.
For instances without batteries, shift move.
For much larger instances, mixed move.

Concluding remarks

- GRASP very fast but produces low quality solutions;
- several moves for Tabu Search;
- trade-off between solution's quality and computing time;
- best results when embedding reduced MILPs in the heuristic.

We achieved producing high quality solutions in short time.
Tabu Search with MILP-zeros move is the best compromise.
For instances without batteries, shift move.
For much larger instances, mixed move.

Concluding remarks

- GRASP very fast but produces low quality solutions;
- several moves for Tabu Search;
- trade-off between solution's quality and computing time;
- best results when embedding reduced MILPs in the heuristic.

We achieved producing high quality solutions in short time.
Tabu Search with MILP-zeros move is the best compromise.
For instances without batteries, shift move.
For much larger instances, mixed move.

Concluding remarks

- GRASP very fast but produces low quality solutions;
- several moves for Tabu Search;
- trade-off between solution's quality and computing time;
- best results when embedding reduced MILPs in the heuristic.

We achieved producing high quality solutions in short time.
Tabu Search with MILP-zeros move is the best compromise.
For instances without batteries, shift move.
For much larger instances, mixed move.

Concluding remarks

- GRASP very fast but produces low quality solutions;
- several moves for Tabu Search;
- trade-off between solution's quality and computing time;
- best results when embedding reduced MILPs in the heuristic.

We achieved producing high quality solutions in short time.
Tabu Search with MILP-zeros move is the best compromise.
For instances without batteries, shift move.
For much larger instances, mixed move.

Future work

- Allowing energy flow among houses;
- optimizing over several days;
- allowing users to change starting windows;
- using non-flat ideal curves.

Future work

- Allowing energy flow among houses;
- optimizing over several days;
- allowing users to change starting windows;
- using non-flat ideal curves.

Future work

- Allowing energy flow among houses;
- optimizing over several days;
- allowing users to change starting windows;
- using non-flat ideal curves.

Future work

- Allowing energy flow among houses;
- optimizing over several days;
- allowing users to change starting windows;
- using non-flat ideal curves.

References I

A. Agnetis, G. Dellino, P. Detti, G. Innocenti, G. de Pascale, and A. Vicino. Appliance Operation Scheduling for Electricity Consumption Optimization. In *50th IEEE Conference on Decision and Control and European Control Conference*, pages 5899–5904, Orlando, Florida, 2011. ISBN 9781612847993. URL http://www.nt.ntnu.no/users/skoge/prost/proceedings/cdc-ecc-2011/data/papers/0506.pdf.

A. Barbato, A. Capone, G. Carello, M. Delfanti, M. Merlo, and A. Zaminga. Cooperative and Non-Cooperative house energy optimization in a Smart Grid perspective. In *2011 IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks*, pages 1–6. IEEE, June 2011a. ISBN 978-1-4577-0352-2. doi: 10.1109/WoWMoM.2011.5986478. URL http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5986478.

References II

A. Barbato, A. Capone, G. Carello, M. Delfanti, M. Merlo, and A. Zaminga. House energy demand optimization in single and multi-user scenarios. In *2011 IEEE International Conference on Smart Grid Communications (SmartGridComm)*, pages 345–350. IEEE, Oct. 2011b. ISBN 978-1-4577-1702-4. doi: 10.1109/SmartGridComm.2011.6102345. URL `http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6102345`.

T. A. Feo and M. G. C. Resende. Greedy Randomized Adaptive Search Procedures. *Journal of Global Optimization*, 6(2): 109–133, Mar. 1995. ISSN 0925-5001. doi: 10.1007/BF01096763. URL `http://www.springerlink.com/index/10.1007/BF01096763`.

F. Glover and M. Laguna. *Tabu search*. Springer, 1998. ISBN 9780792381877. URL `http://www.amazon.com/exec/obidos/ASIN/079239965X/ref=nosim/weisstein-20`.

References III

S. Kishore and L. V. Snyder. Control Mechanisms for Residential Electricity Demand in SmartGrids. In *2010 First IEEE International Conference on Smart Grid Communications*, pages 443–448. IEEE, Oct. 2010. ISBN 978-1-4244-6510-1. doi: 10.1109/SMARTGRID.2010.5622084. URL http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5622084.

N. Kowahl and A. Kuh. Micro-scale smart grid optimization. In *The 2010 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, July 2010. ISBN 978-1-4244-6916-1. doi: 10.1109/IJCNN.2010.5596726. URL http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5596726.

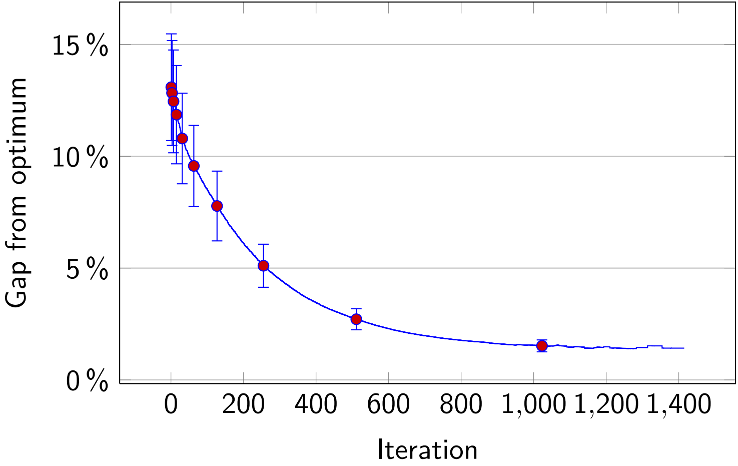
References IV

D. Livengood and R. Larson. The Energy Box: Locally Automated Optimal Control of Residential Electricity Usage. *Service Science*, 1(1):1–16, Mar. 2009. ISSN 2164-3962. doi: 10.1287/serv.1.1.1. URL http://servsci.journal.informs.org/cgi/doi/10.1287/serv.1.1.1.

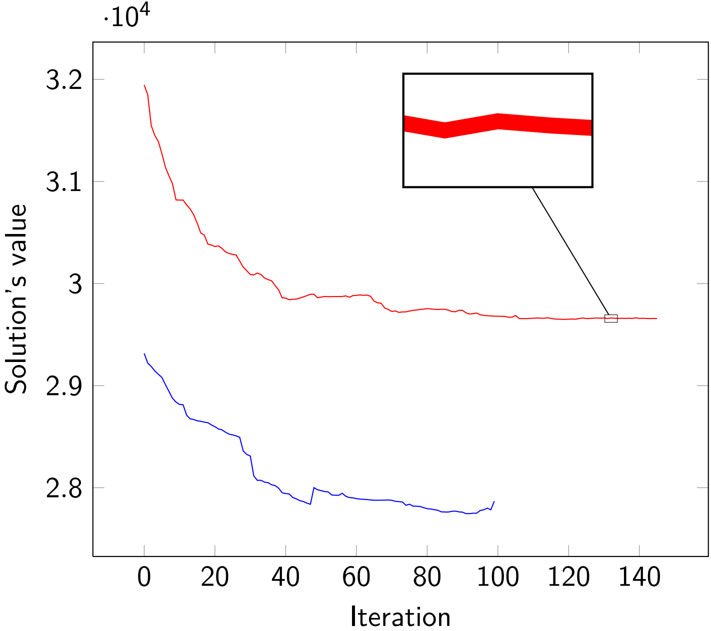
A.-H. Mohsenian-Rad and A. Leon-Garcia. Optimal Residential Load Control With Price Prediction in Real-Time Electricity Pricing Environments. *IEEE Transactions on Smart Grid*, 1(2): 120–133, Sept. 2010. ISSN 1949-3053. doi: 10.1109/TSG.2010.2055903. URL http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5540263.

Tabu Search trend

Average Tabu Search trend for 400 house, 0 batteries instances.
Early iterations result in larger improvements.



Tabu Moves usage



Tabu Search results

	20 houses		200 houses		400 houses	
Batteries	0	2	0	20	0	40
Shift	5.33%	14.72%	1.64%	11.33%	1.54%	11.31%
Mixed	5.23%	13.93%	1.58%	6.09%	1.45%	5.76%
Mixed MILP	5.23%	12.57%	1.58%	10.58%	1.45%	12.70%
MILP-zeroes	1.67%	2.39%	0.63%	2.40%	1.15%	2.78%
PLR	0.86%	1.76%	0.19%	2.73%	0.12%	3.27%
PLR and TS	-0.07%	1.62%	0.00%	2.29%	0.03%	2.10%
Local Branch.	0.61%	1.69%	0.41%	5.81%	4.47%	13.04%

Instances with batteries are harder to solve. Pure Tabu Search methods produce worst results, mixed Tabu Search and MILP methods produce good results in short time, pure MILP methods produce best results in longer time.

Results - Tabu Search 20 houses

Batteries	No			Yes		
	Gap	Dev	Time	Gap	Dev	Time
Shift	5.33%	2.17%	0.8	14.72%	2.92%	0.87
Mixed	5.23%	2.63%	0.87	13.93%	3.01%	0.87
MILP-zeros	1.67%	1.25%	31.23	2.39%	1.13%	31.53
PLR	0.86%	0.81%	31.0	1.76%	1.21%	31.1
PLR and TS	-0.07%	0.9%	61.0	1.62%	1.13%	60.3
Local Branc.	0.61%	0.86%	92.53	1.69%	1.34%	92.87

Generating 10 solutions with GRASP and improving the best one.
Solution's quality and elapsed time.

Results - Tabu Search 200 houses

Batteries	No			Yes		
	Gap	Dev	Time	Gap	Dev	Time
Shift	1.64%	0.4%	24.3	11.33%	1.15%	24.57
Mixed	1.58%	0.39%	26.2	6.09%	2.36%	34.93
MILP-zeros	0.63%	0.24%	40.87	2.40%	0.81%	79.7
PLR	0.19%	0.16%	76.13	2.73%	1.18%	88.63
PLR and TS	0.00%	0.16%	138.14	2.29%	1.0%	147.56
Local Branc.	0.41%	0.19%	79.3	5.81%	5.92%	1444.24

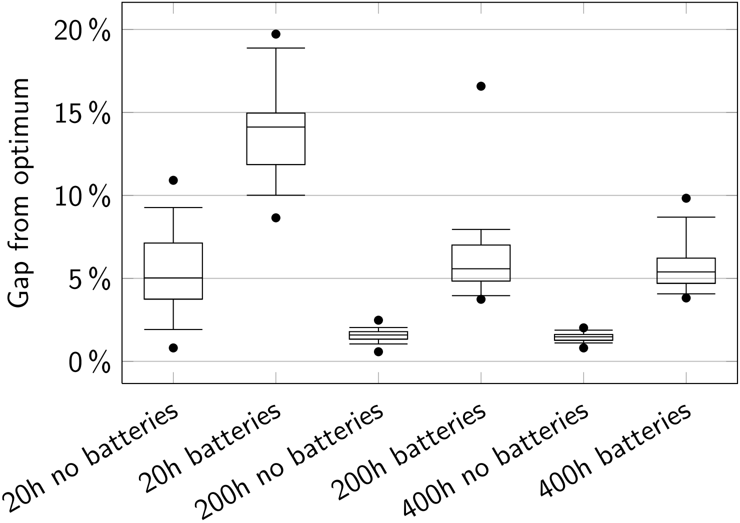
Generating 10 solutions with GRASP and improving the best one.
Solution's quality and elapsed time.

Results - Tabu Search 400 houses

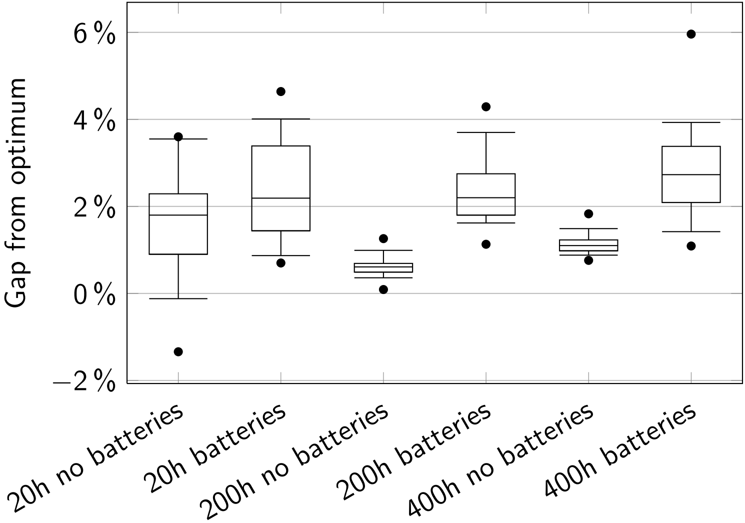
Batteries		No			Yes	
	Gap	Dev	Time	Gap	Dev	Time
Shift	1.54%	0.29%	78.07	11.31%	0.94%	79.03
Mixed	1.45%	0.28%	89.73	5.76%	1.49%	122.2
MILP-zeros	1.15%	0.23%	62.8	2.78%	0.97%	177.43
PLR	0.12%	0.09%	188.67	3.27%	2.25%	227.1
PLR and TS	0.03%	0.1%	342.93	2.10%	1.19%	384.36
Local Branc.	4.47%	0.97%	1772.3	13.04%	1.07%	2627.4

Generating 10 solutions with GRASP and improving the best one.
Solution's quality and elapsed time.

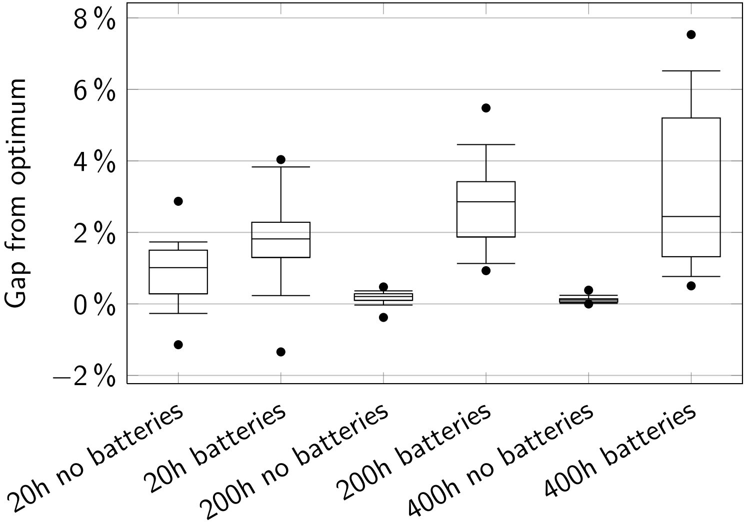
Mixed move with battery moves



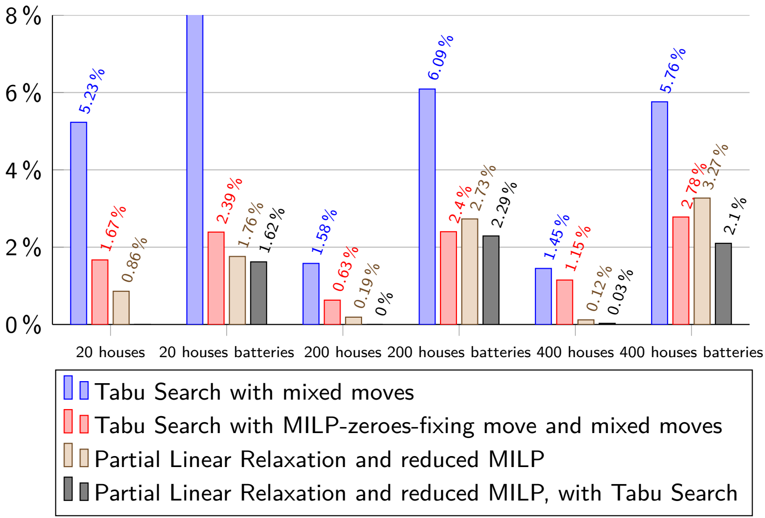
Mixed move with MILP-zeroes-fixing move



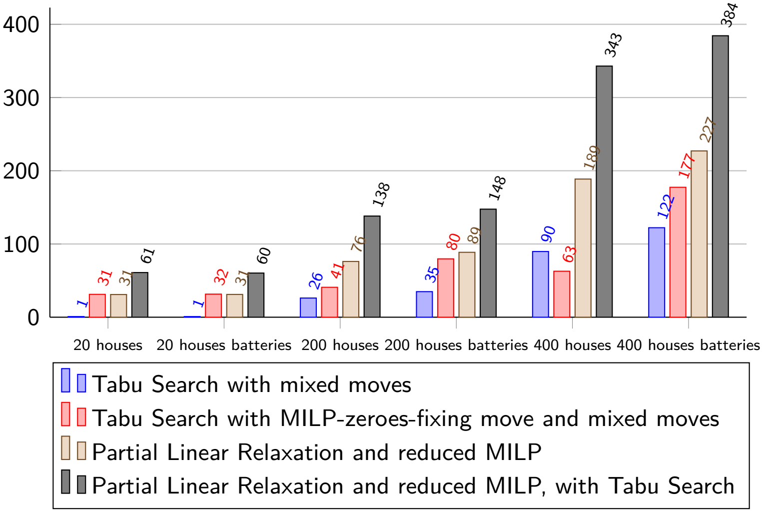
Partial linear relaxation and reduced MILP



Comparison - Gap from reference



Comparison - Computing time



Strategic oscillation

- Release the local maximal peak constraint to cross the infeasible region;
- *slacks* measure house's infeasibility:

$$s_h \triangleq \max\left(0, \max_t y_{h,t} - \pi_t^{\text{in}}\right) \quad \forall h \in H$$
$$s \triangleq \sum_{h \in H} s_h;$$

- restrict the neighbourhood to solutions that reduce slacks to recover feasibility.

Strategic oscillation

- Release the local maximal peak constraint to cross the infeasible region;
- *slacks* measure house's infeasibility:

$$
\begin{aligned}
s_h &\triangleq \max\left(0, \max_t y_{h,t} - \pi_t^{\text{in}}\right) \quad \forall h \in H \\
s &\triangleq \sum_{h \in H} s_h;
\end{aligned}
$$

- restrict the neighbourhood to solutions that reduce slacks to recover feasibility.

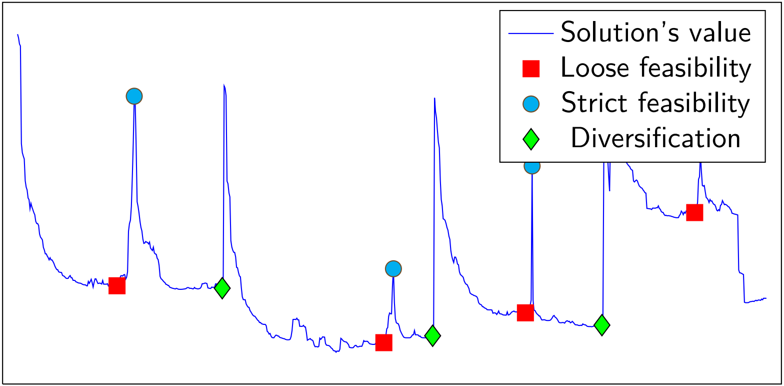
Strategic oscillation

- Release the local maximal peak constraint to cross the infeasible region;
- *slacks* measure house's infeasibility:

$$s_h \triangleq \max\left(0, \max_t y_{h,t} - \pi_t^{\text{in}}\right) \quad \forall h \in H$$
$$s \triangleq \sum_{h \in H} s_h;$$

- restrict the neighbourhood to solutions that reduce slacks to recover feasibility.

Infeasible exploration



Local branching

- Iteratively solving a reduced MILP problem;

Distance between two solutions

$\Delta(x, y) = \dots$

- A reduced MILP problem is solved, excluding solutions farther from the current one, i. e., imposing $\Delta(x, x_k) \leq m$;
- If the reduced problem is optimally solved or infeasible the whole region is excluded from future exploration, i. e., imposing $\Delta(x, x_k) \geq m + 1$;
- Otherwise, if a solution x_{k+1} was found only that is excluded from future exploration, i. e., imposing $\Delta(x, x_{k+1}) \geq 1$;
- Otherwise, either m increases or a diversification occurs.

Local branching

- Iteratively solving a reduced MILP problem;

Distance between two solutions

$\Delta(x, y) = \dots$

- A reduced MILP problem is solved, excluding solutions farther from the current one, i. e., imposing $\Delta(x, x_k) \leq m$;
- If the reduced problem is optimally solved or infeasible the whole region is excluded from future exploration, i. e., imposing $\Delta(x, x_k) \geq m + 1$;
- Otherwise, if a solution x_{k+1} was found only that is excluded from future exploration, i. e., imposing $\Delta(x, x_{k+1}) \geq 1$;
- Otherwise, either m increases or a diversification occurs.

Local branching

- Iteratively solving a reduced MILP problem;

Distance between two solutions

$\Delta(x, y) = \dots$

- A reduced MILP problem is solved, excluding solutions farther from the current one, i. e., imposing $\Delta(x, x_k) \leq m$;
- If the reduced problem is optimally solved or infeasible the whole region is excluded from future exploration, i. e., imposing $\Delta(x, x_k) \geq m + 1$;
- Otherwise, if a solution x_{k+1} was found only that is excluded from future exploration, i. e., imposing $\Delta(x, x_{k+1}) \geq 1$;
- Otherwise, either m increases or a diversification occurs.

Local branching

- Iteratively solving a reduced MILP problem;

Distance between two solutions

$\Delta(x, y) = \dots$

- A reduced MILP problem is solved, excluding solutions farther from the current one, i. e., imposing $\Delta(x, x_k) \leq m$;
- If the reduced problem is optimally solved or infeasible the whole region is excluded from future exploration, i. e., imposing $\Delta(x, x_k) \geq m + 1$;
- Otherwise, if a solution x_{k+1} was found only that is excluded from future exploration, i. e., imposing $\Delta(x, x_{k+1}) \geq 1$;
- Otherwise, either m increases or a diversification occurs.

Local branching

- Iteratively solving a reduced MILP problem;

Distance between two solutions

$\Delta(x, y) = \ldots$

- A reduced MILP problem is solved, excluding solutions farther from the current one, i. e., imposing $\Delta(x, x_k) \leq m$;
- If the reduced problem is optimally solved or infeasible the whole region is excluded from future exploration, i. e., imposing $\Delta(x, x_k) \geq m + 1$;
- Otherwise, if a solution x_{k+1} was found only that is excluded from future exploration, i. e., imposing $\Delta(x, x_{k+1}) \geq 1$;
- Otherwise, either m increases or a diversification occurs.

Local branching

- Iteratively solving a reduced MILP problem;

Distance between two solutions

$\Delta(x, y) = \dots$

- A reduced MILP problem is solved, excluding solutions farther from the current one, i. e., imposing $\Delta(x, x_k) \leq m$;
- If the reduced problem is optimally solved or infeasible the whole region is excluded from future exploration, i. e., imposing $\Delta(x, x_k) \geq m + 1$;
- Otherwise, if a solution x_{k+1} was found only that is excluded from future exploration, i. e., imposing $\Delta(x, x_{k+1}) \geq 1$;
- Otherwise, either m increases or a diversification occurs.