Decision aid methodologies in transportation

Lecture 3: Crew Scheduling

Prem Kumar

prem.viswanathan@epfl.ch

Transport and Mobility Laboratory

TRANSP-OR

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

This course is an extension of the same course taught last year by Dr Niklaus Eggenberg.
A few slides are inspired from the material used by him and Prof C Barnhart (MIT Courseware)

Summary

- We learnt about the different aircraft scheduling models
- We learnt to formulate these sub-problems into mathematical models
- We learnt to solve certain problems with heuristics
- However heuristics have their limitations
- Today we will learn about exhaustive enumeration methods
- And of course, crew scheduling problems – crew pairing and crew rostering

Review: Maintenance Routing Formulation

$$\text{Minimize} \sum_{r \in R} c_r x_r + \sum_{f \in F} c_f y_f$$

Subject to:

$$\sum_{r \in R} b_{r,f} x_r + y_f = 1, \ \forall f \in F \qquad (1)$$

$$\sum_{r \in R} b_{r,p} x_r \leq 1, \ \forall p \in P \qquad (2)$$

Bounds

$$x_r \in \{0,1\}$$

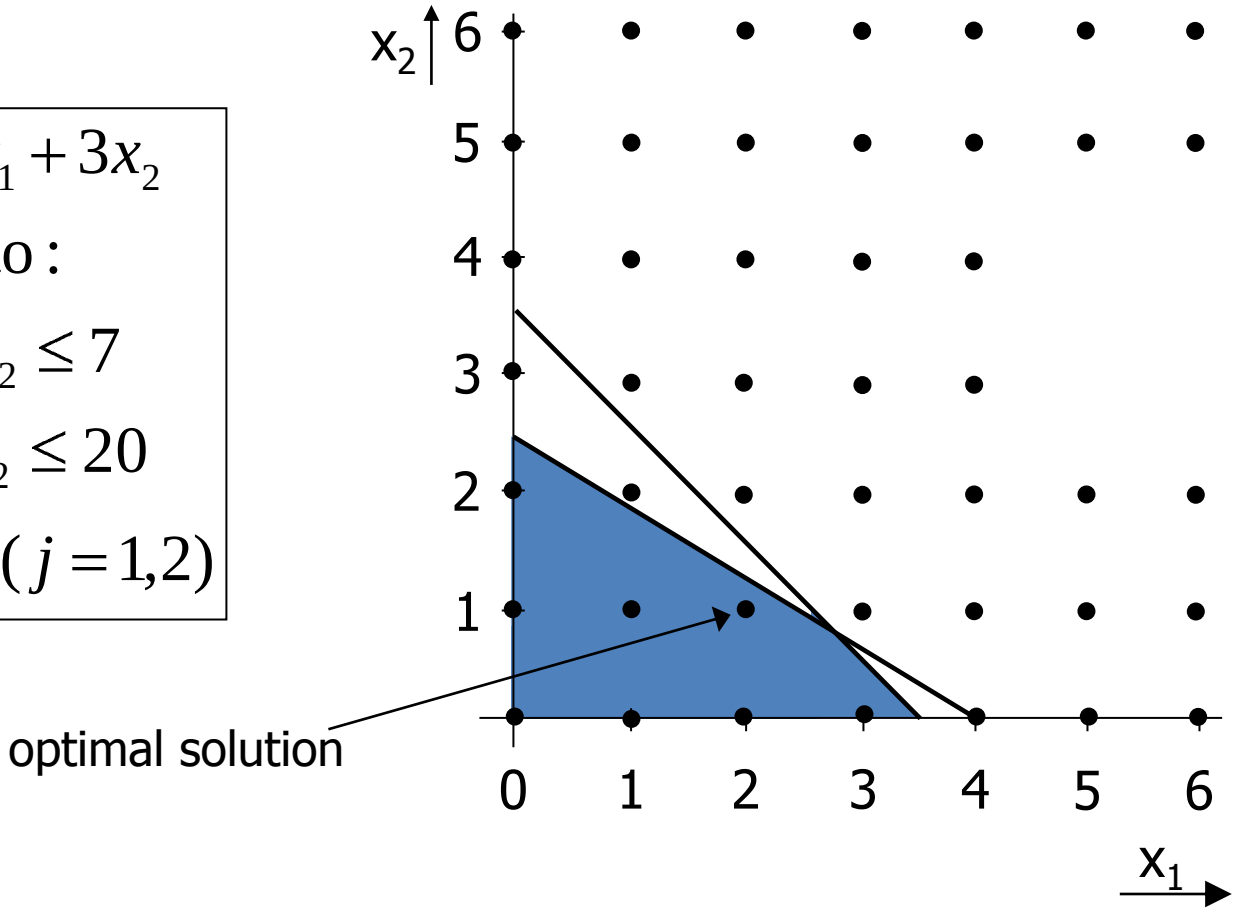
$$y_f \in \{0,1\}$$

Mixed Integer Programming Models

- If only the x_r and y_f variables are continuous instead of discrete, the model could have been solved with simplex algorithm
- Unlike linear programming, optimal solution is not guaranteed to be in the corner or peripheral points of the feasible region
- These type of formulations are referred to as Mixed Integer Programming (MIP) models

Mixed Integer Programming Models

$$\max \quad 2x_1 + 3x_2$$
$$\text{subject to:}$$
$$2x_1 + 2x_2 \leq 7$$
$$5x_1 + 8x_2 \leq 20$$
$$x_j \in Z^+ \ (j = 1,2)$$



Exhaustive Enumeration: Branch and Bound

- As discussed in the first session, many practical problems can be represented through a tree
- This tree can be navigated further down till either you find an optimal solution or conclude that there are no more feasible solutions
- Branch and Bound algorithm is a specialized tree search technique used commonly to solve MIPs
- Using B&B, we can solve both Fleet Assignment Model as well as Maintenance Routing Model

Optimization Terminology

- For a *minimization* problem
  - Upper Bound = value for which we know at least one solution exists
  - Lower Bound = value for which we know that no solution with lower value exists
  - Optimality gap = (UB-LB)/LB
- For a *maximization* problem
  - LB = value for which we know at least one solution exists
  - UB = value for which we know that no solution with higher value exists
  - Optimality gap = (UB-LB)/LB

Branch and Bound Methodology

- Step 1: Relax integrality constraints

  $x_j \in Z^+ \Rightarrow x_j \geq 0$

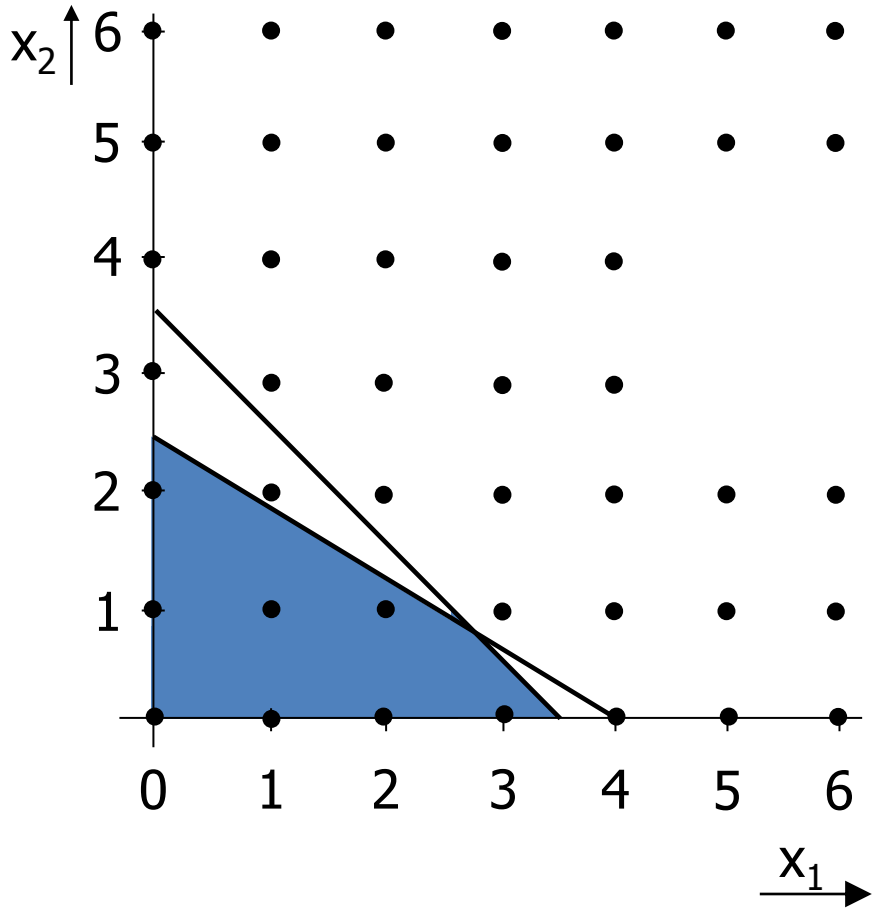
  $x_j \in \{0, 1\} \Rightarrow [0, 1] \Rightarrow 0 \leq x_j \leq 1$

- Step 2: Solve relaxed problem
- Step 3: Branch-and-bound loop
  - Branch: redefine bounds for integral variables with non-integral solutions or fix binary variables
  - Bound: use non-integral objective function value as lower bound

    use integral solution as an upper bound
  - Stopping criteria: If current LB = UB or every node is evaluated

Branch and Bound Example

$$\max \quad 2x_1 + 3x_2$$
$$\text{subject to:}$$
$$2x_1 + 2x_2 \leq 7$$
$$5x_1 + 8x_2 \leq 20$$
$$x_j \in Z^+ \ (j = 1,2)$$



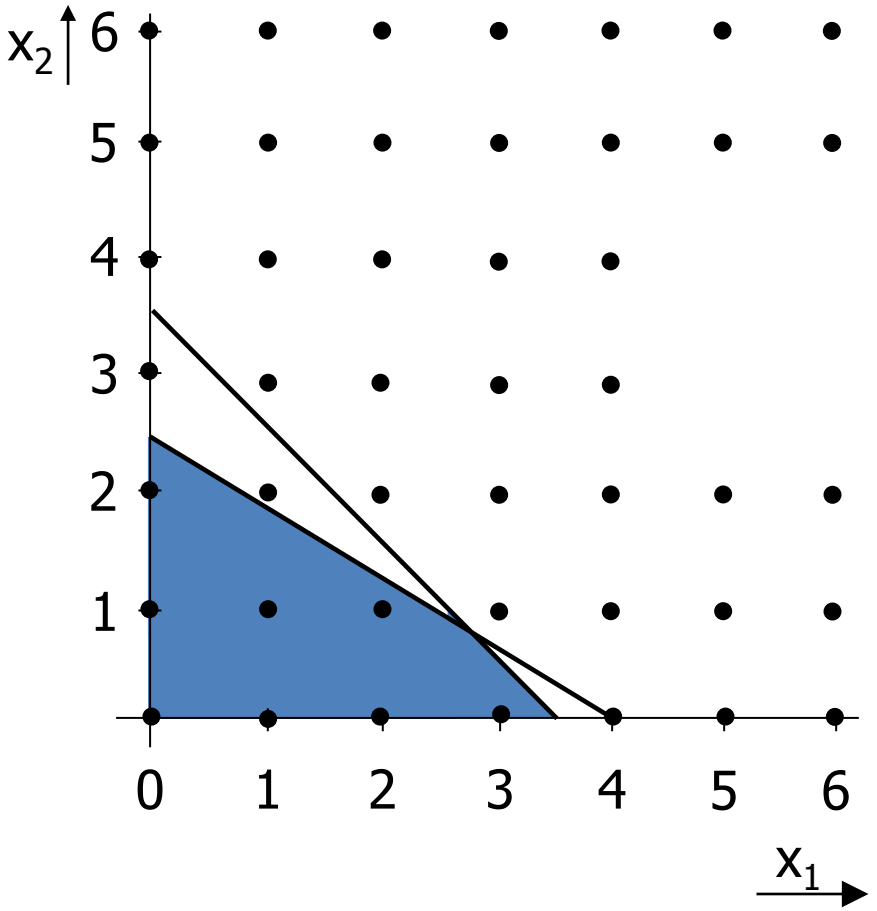
Branch and Bound: Solve the Relaxed Problem

$$\max \quad 2x_1 + 3x_2$$
$$\text{subject to:}$$
$$2x_1 + 2x_2 \le 7$$
$$5x_1 + 8x_2 \le 20$$
$$x_j \ge 0 \; (j = 1,2)$$

OptimalSolution

$$x_1 = \frac{8}{3}$$

$$x_2 = \frac{5}{6}$$



Branch and Bound: Branching Option

$$\begin{aligned}
&\max \quad 2x_1 + 3x_2 \\
&\text{subject to:} \\
&2x_1 + 2x_2 \le 7 \\
&5x_1 + 8x_2 \le 20 \\
&x_j \ge 0 \; (j = 1,2)
\end{aligned}$$

OptimalSolution

$$x_1 = \frac{8}{3}$$

$$x_2 = \frac{5}{6}$$

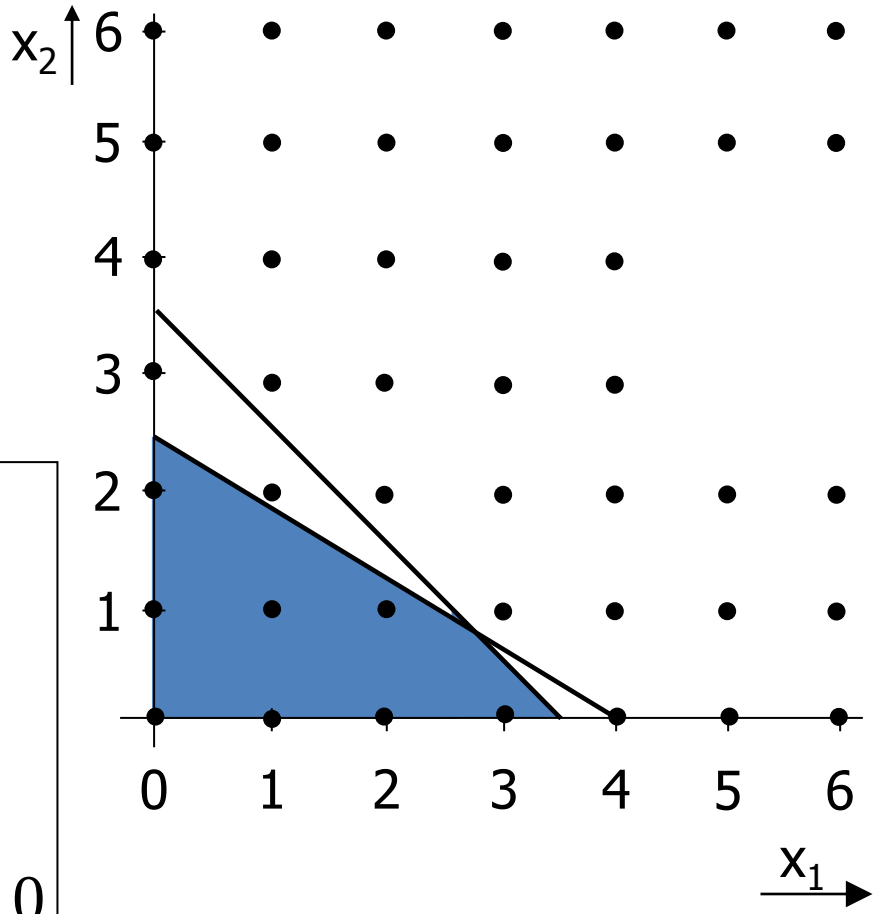
Branching

$$x_1 \ge 3$$

$$x_1 \le 2$$

$$x_2 \ge 1$$

$$x_2 \le 0 \Rightarrow x_2 = 0$$



Branch and Bound: Compute Bounds

$$\begin{aligned}
&\max \quad 2x_1 + 3x_2 \\
&\text{subject to:} \\
&2x_1 + 2x_2 \le 7 \\
&5x_1 + 8x_2 \le 20 \\
&x_j \ge 0 \; (j = 1,2)
\end{aligned}$$

$$UB = \frac{47}{6}$$

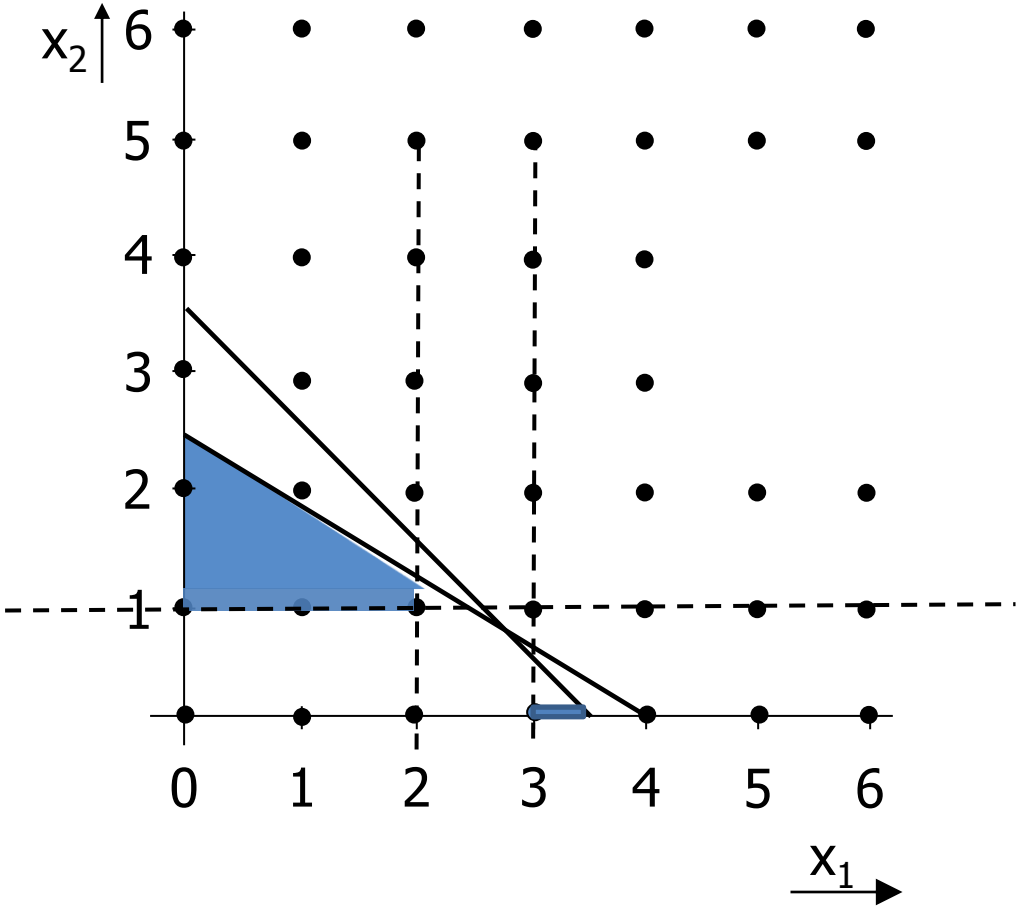
$$LB = 0$$

Branching

$$x_1 \ge 3$$

$$x_1 \le 2$$

$$x_2 \ge 1$$



Branch and Bound: Branching on x_1

$$\max \; 2x_1 + 3x_2$$
$$\text{subject to:}$$
$$2x_1 + 2x_2 \le 7$$
$$5x_1 + 8x_2 \le 20$$
$$x_j \ge 0 \; (j = 1,2)$$

Branching

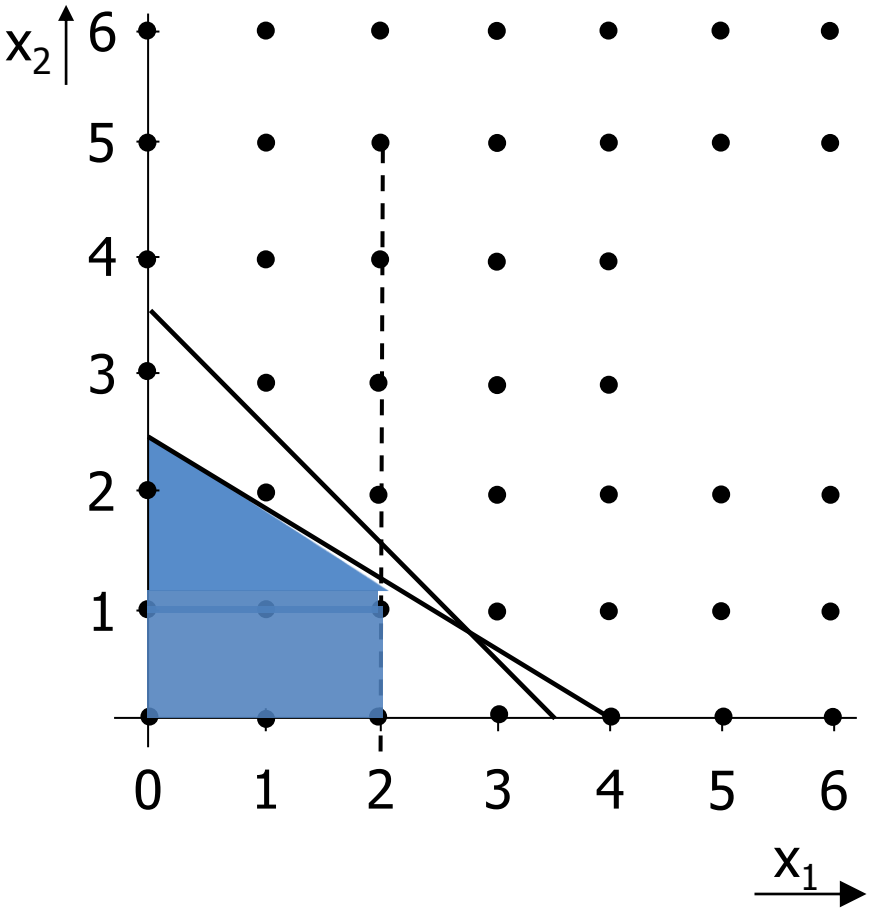
$$x_1 \ge 3$$
$$x_1 \le 2$$
$$x_2 \ge 1$$

$$UB = \frac{47}{6}$$
$$LB = 0$$

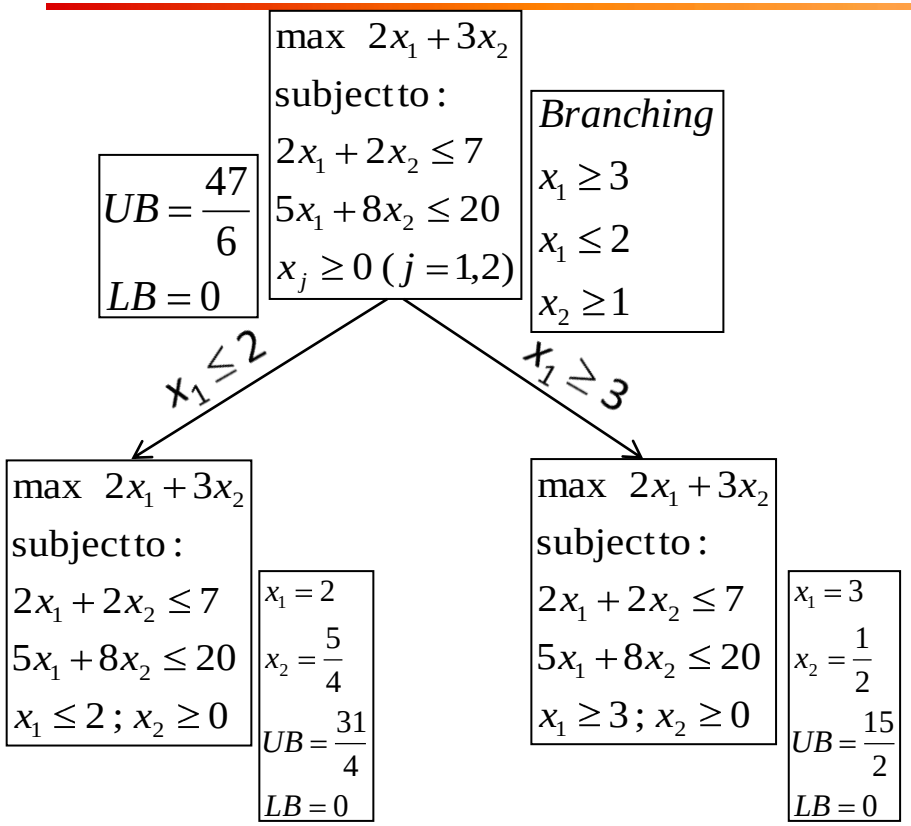
$x_1 \le 2$

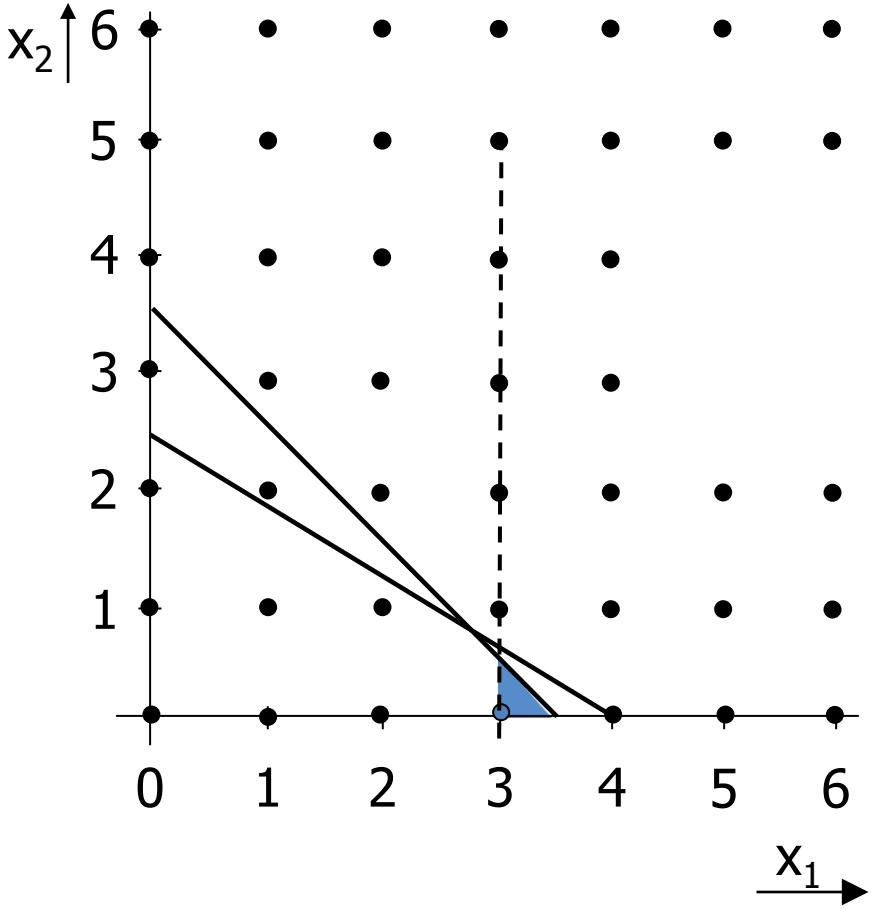
$$\max \; 2x_1 + 3x_2$$
$$\text{subject to:}$$
$$2x_1 + 2x_2 \le 7$$
$$5x_1 + 8x_2 \le 20$$
$$x_1 \le 2 \; ; \; x_2 \ge 0$$

$$x_1 = 2$$
$$x_2 = \frac{5}{4}$$
$$UB = \frac{31}{4}$$
$$LB = 0$$



Branch and Bound: Branching on x_1

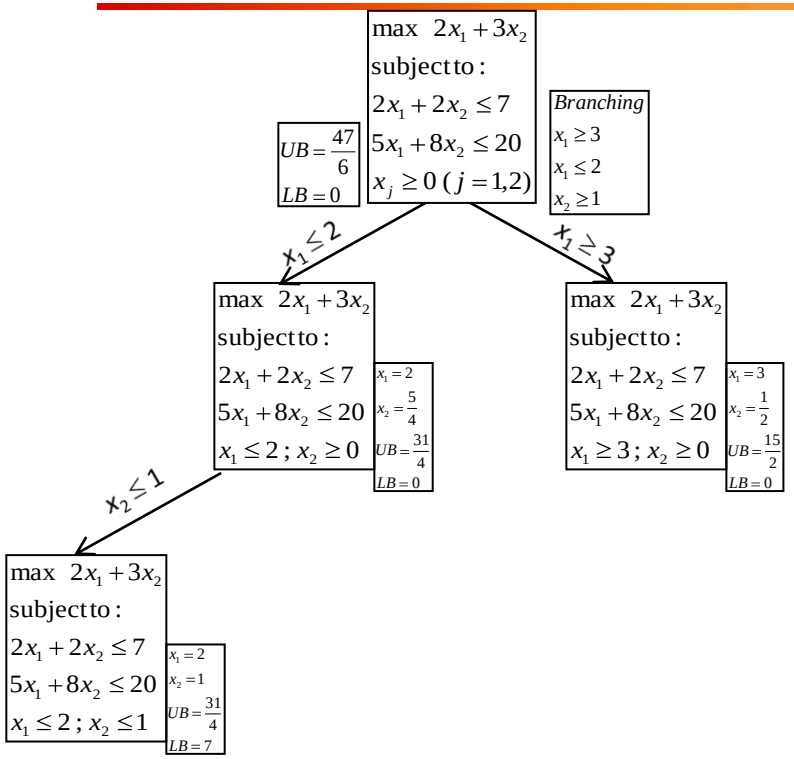


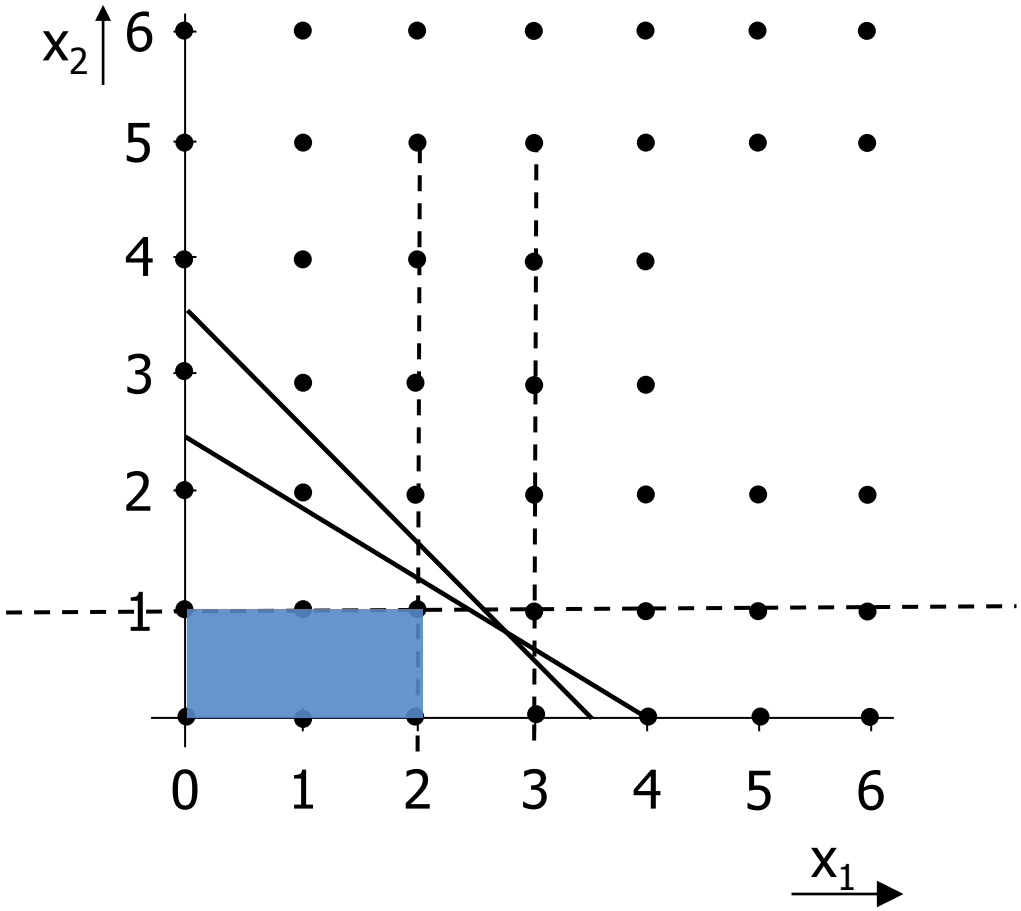


TRANSP-OR

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

Branch and Bound: Branching on x_2

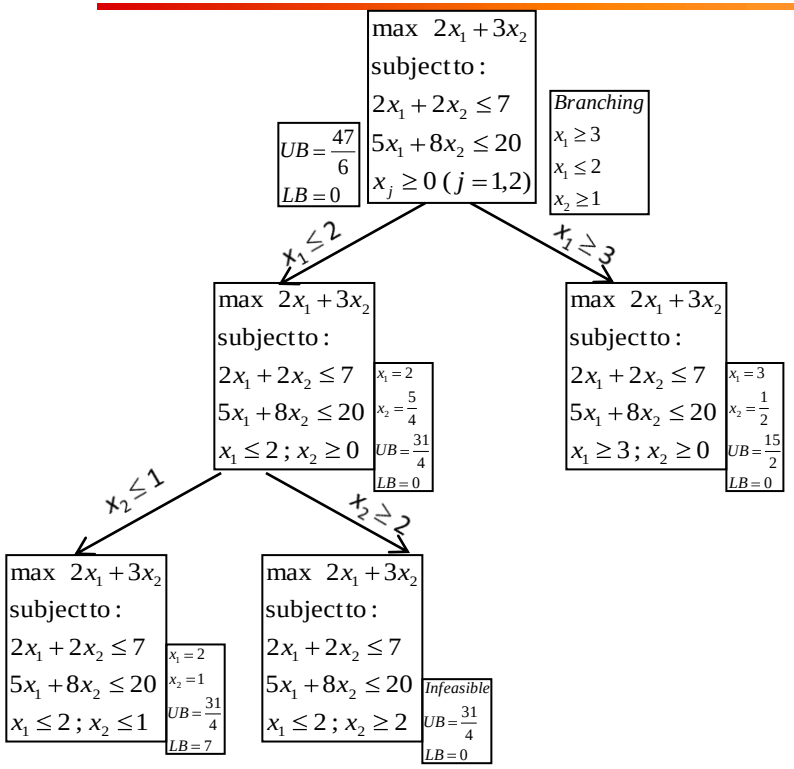


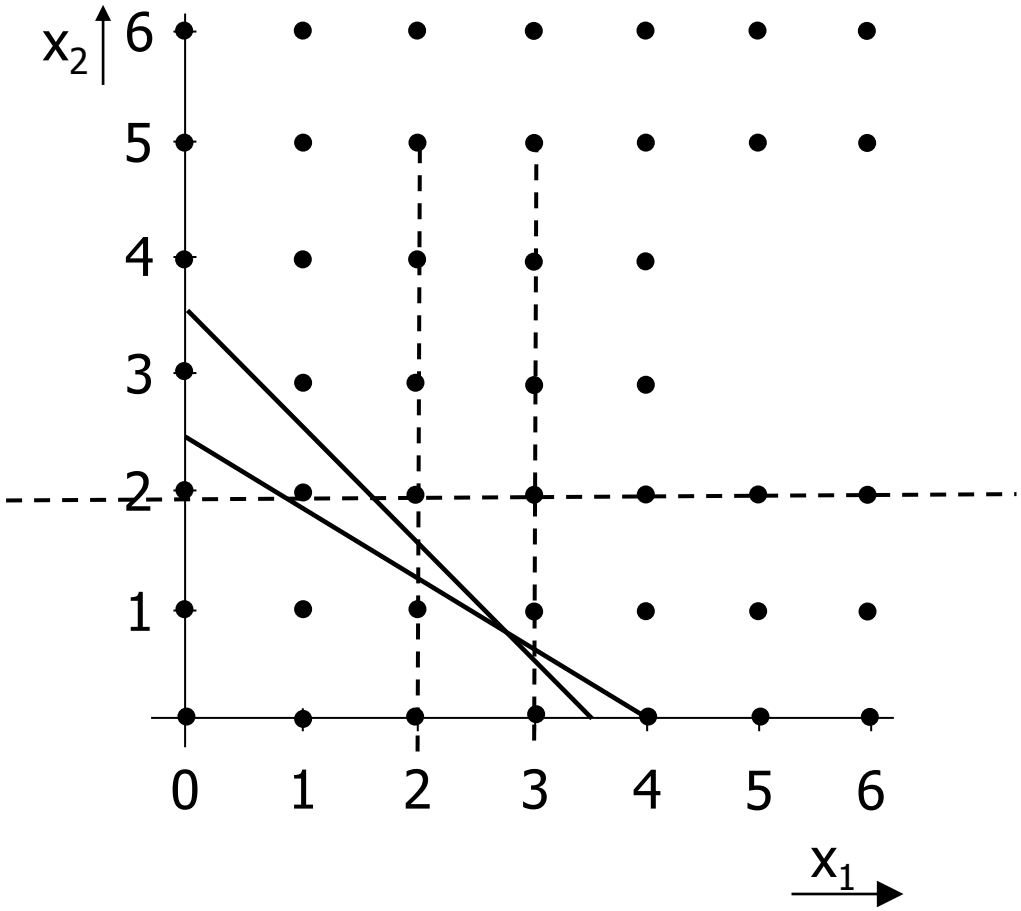


TRANSP-OR

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

Branch and Bound: Branching on x_2

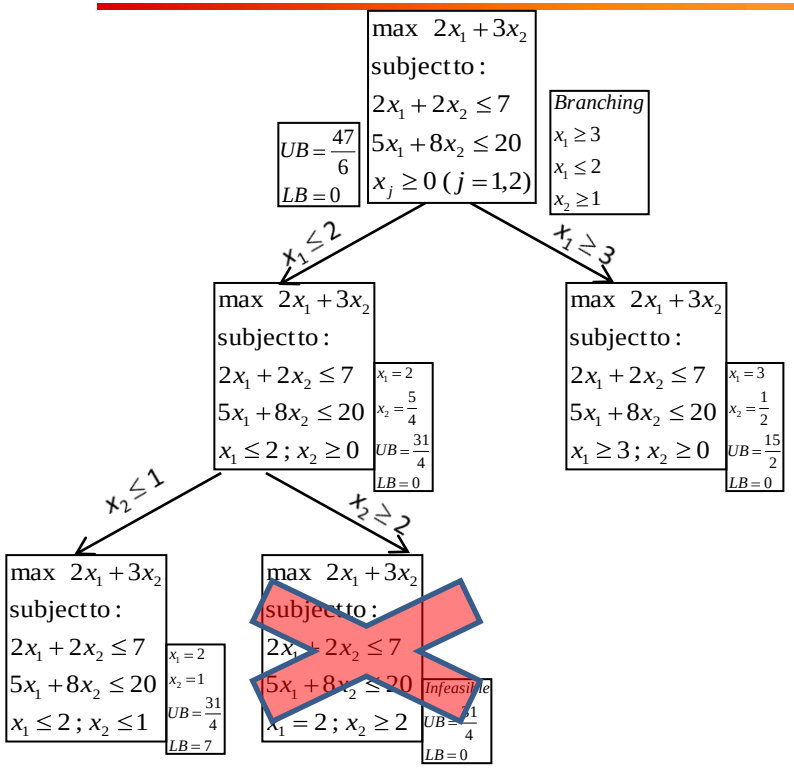


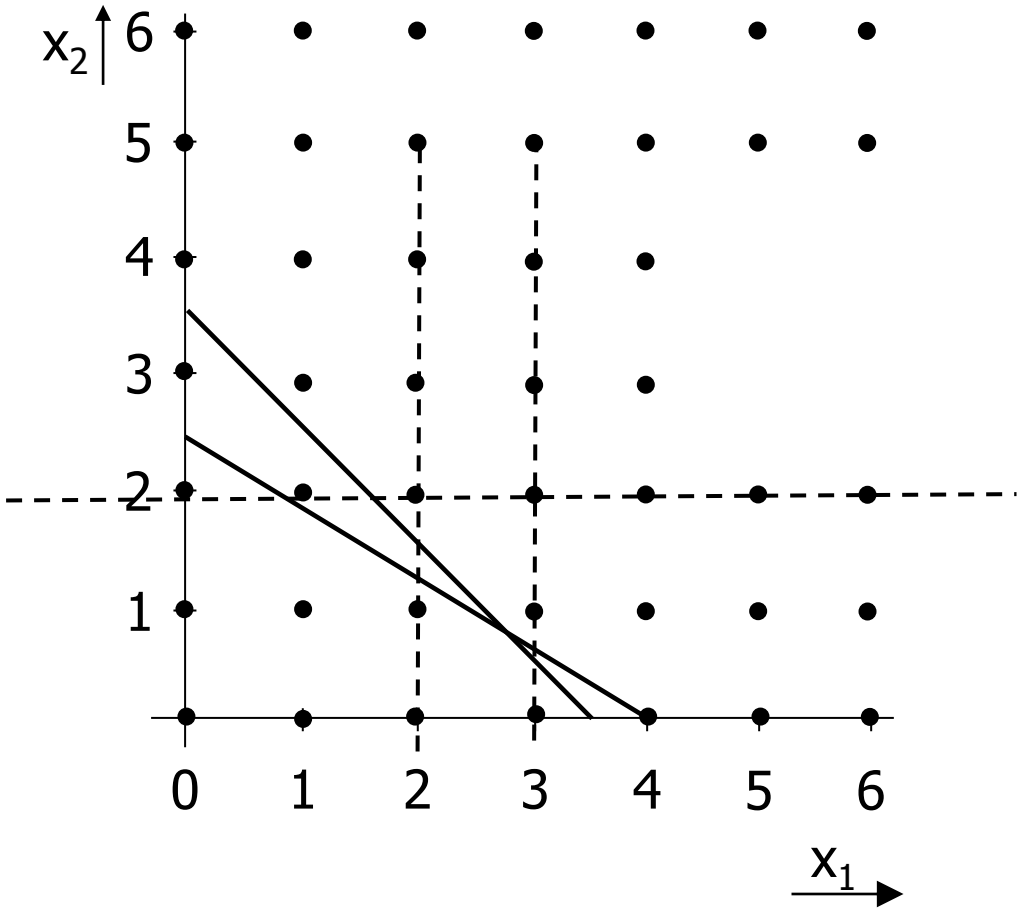


TRANSP-OR

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

Branch and Bound: Branching on x_2

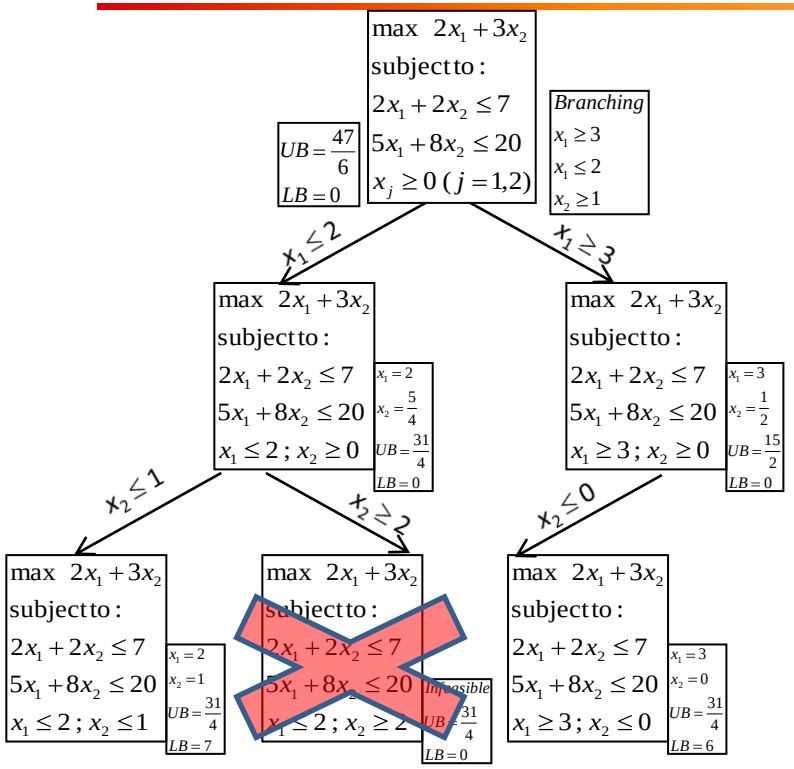


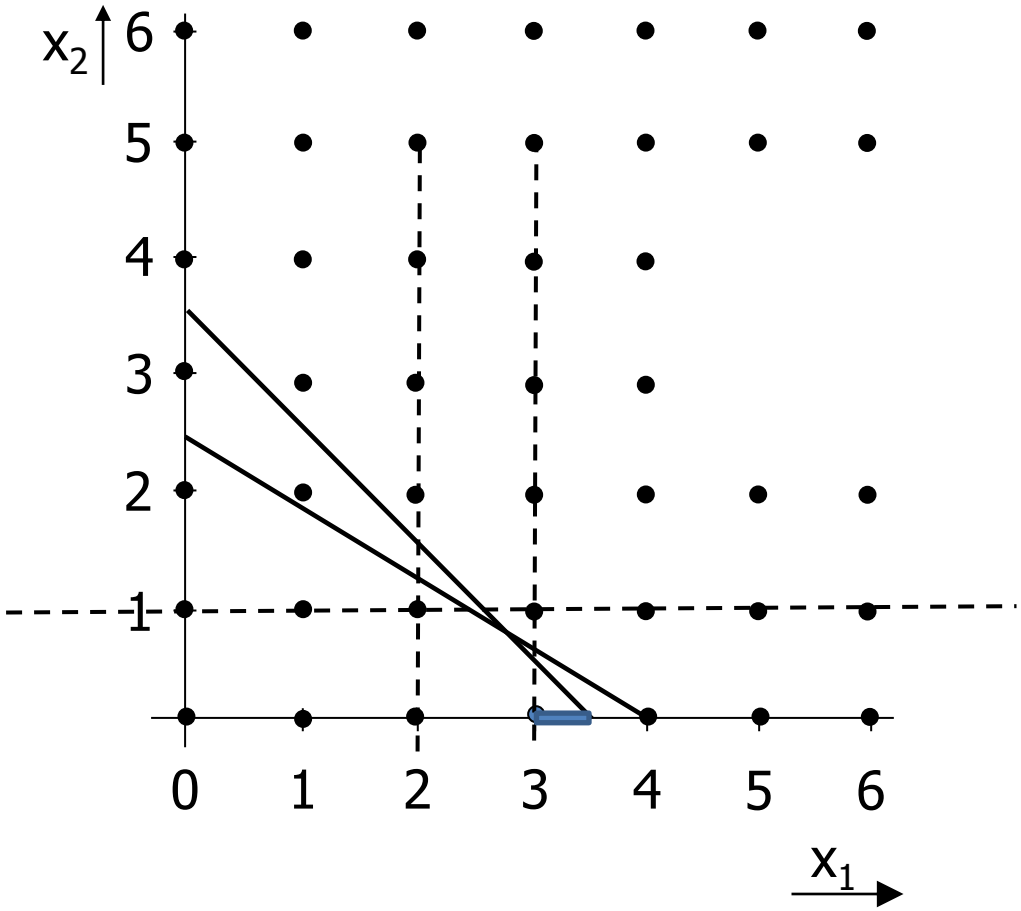


TRANSP-OR

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

Branch and Bound: Branching on x_2

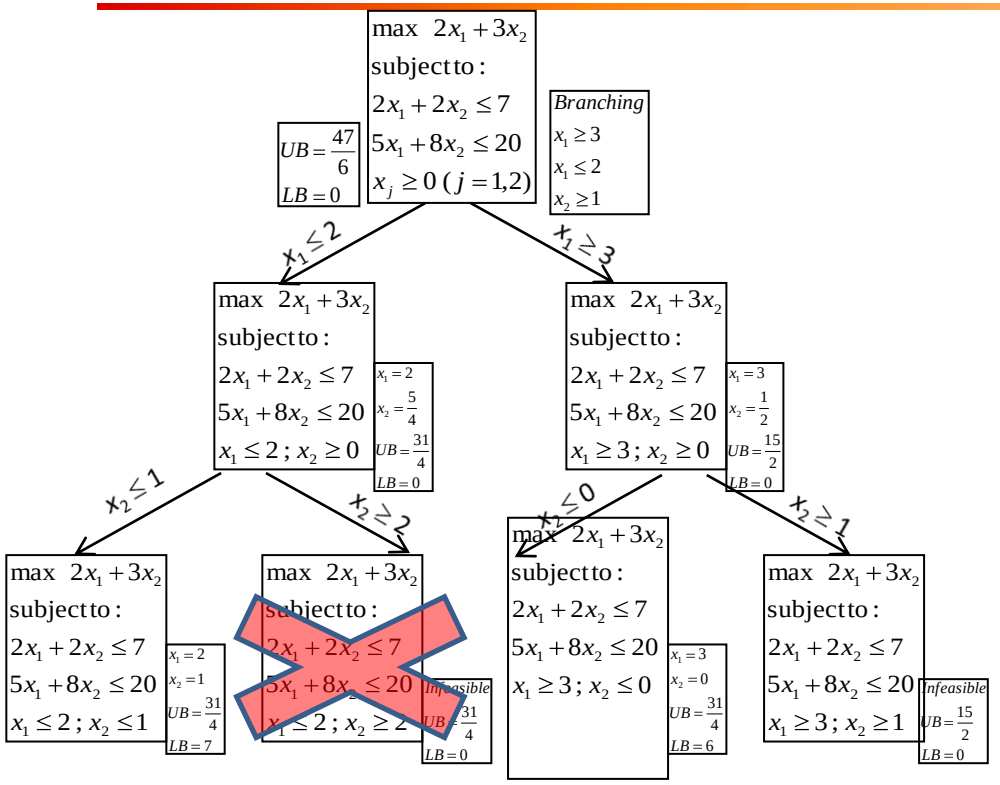


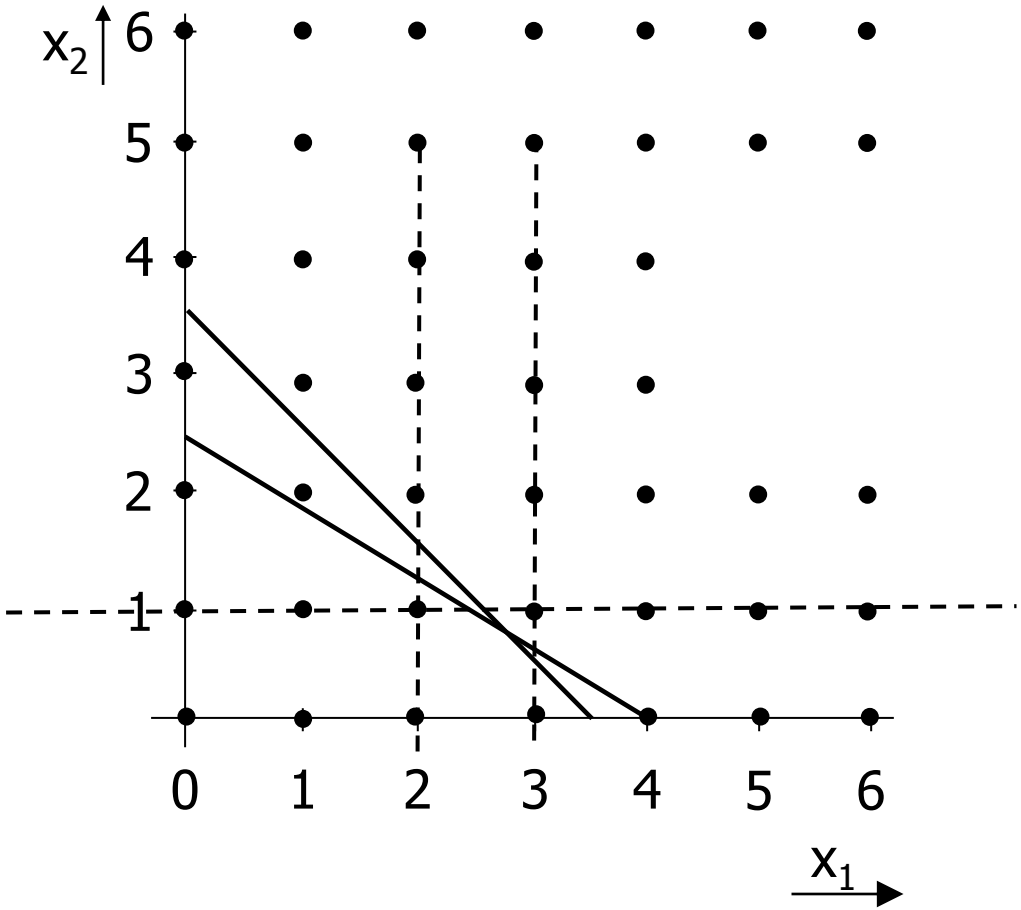


TRANSP-OR

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

Branch and Bound: Branching on x_2

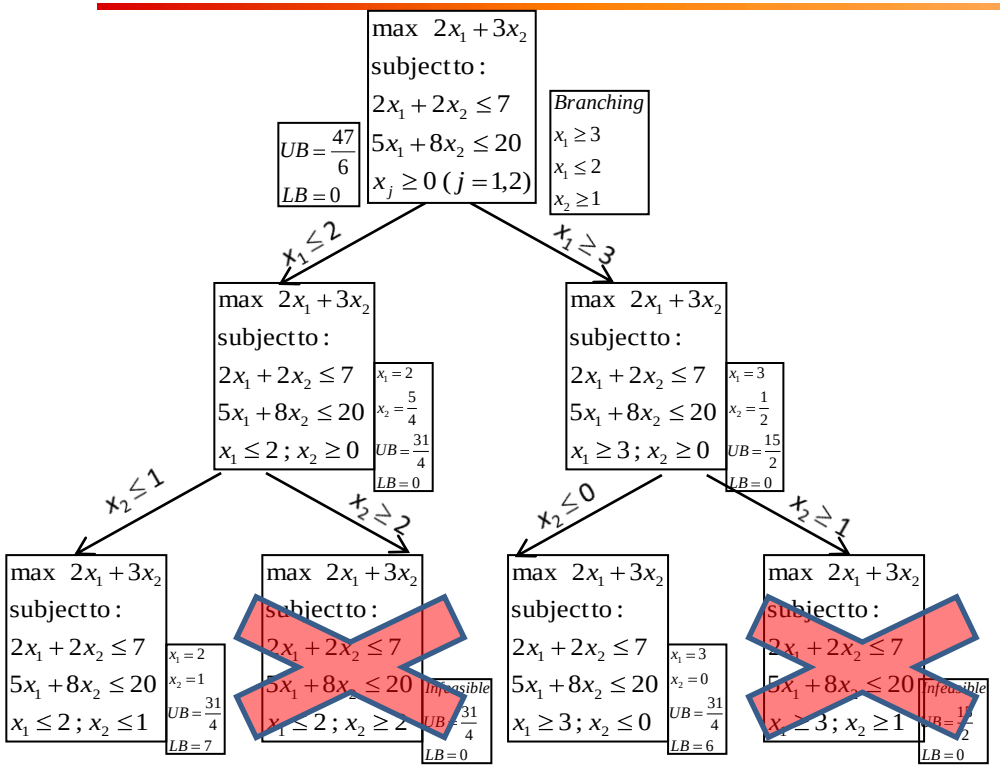


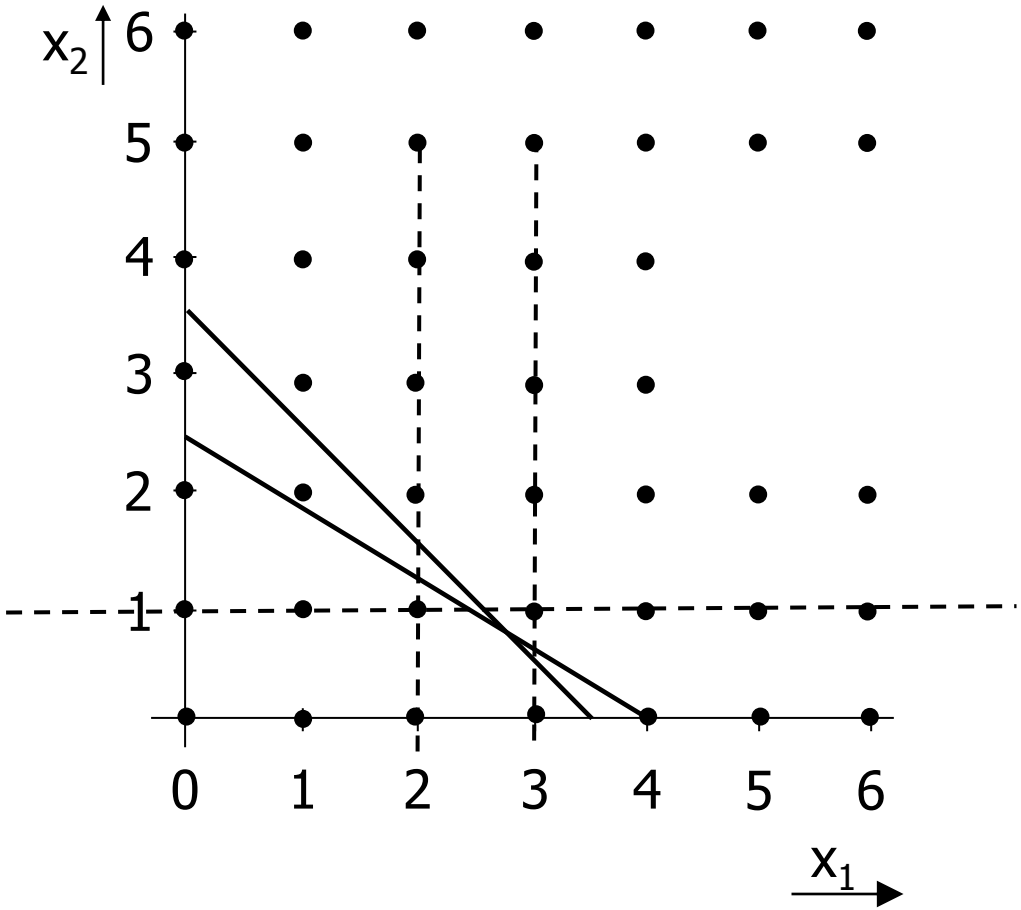


TRANSP-OR

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

Branch and Bound: Branching on x_2

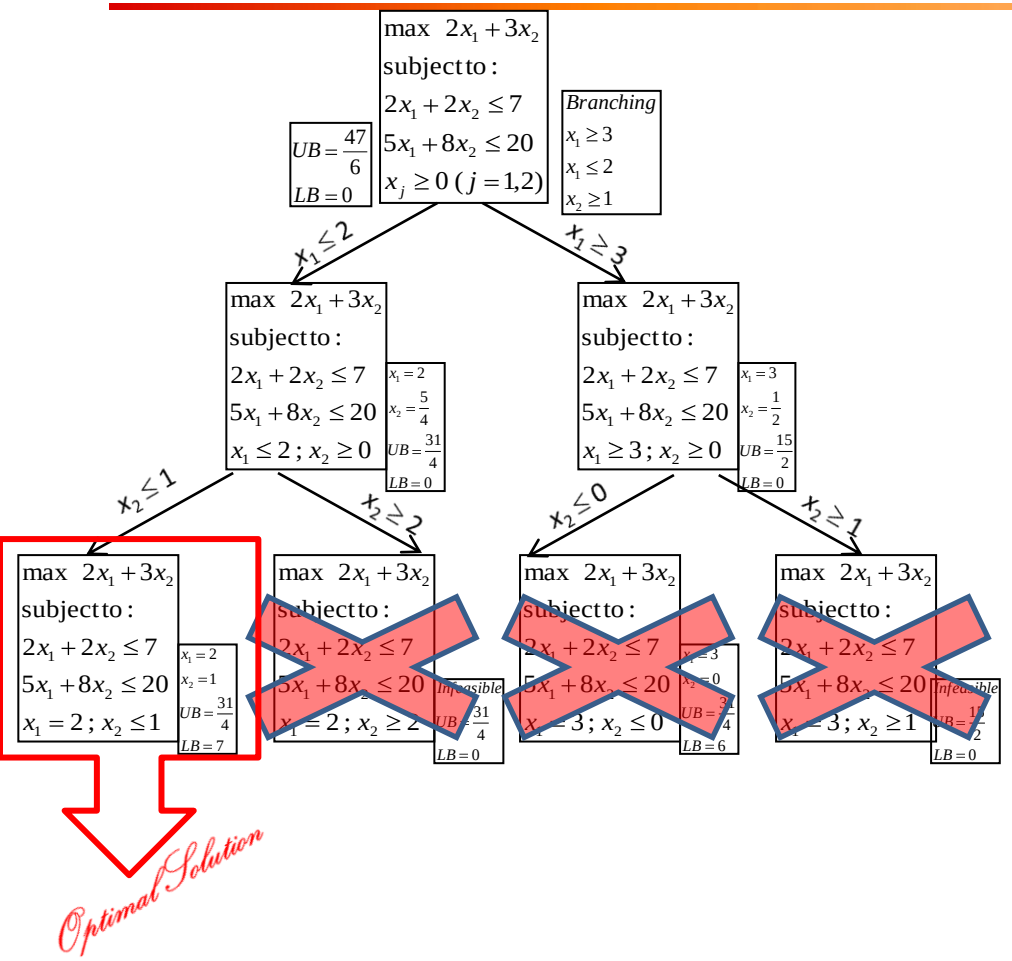


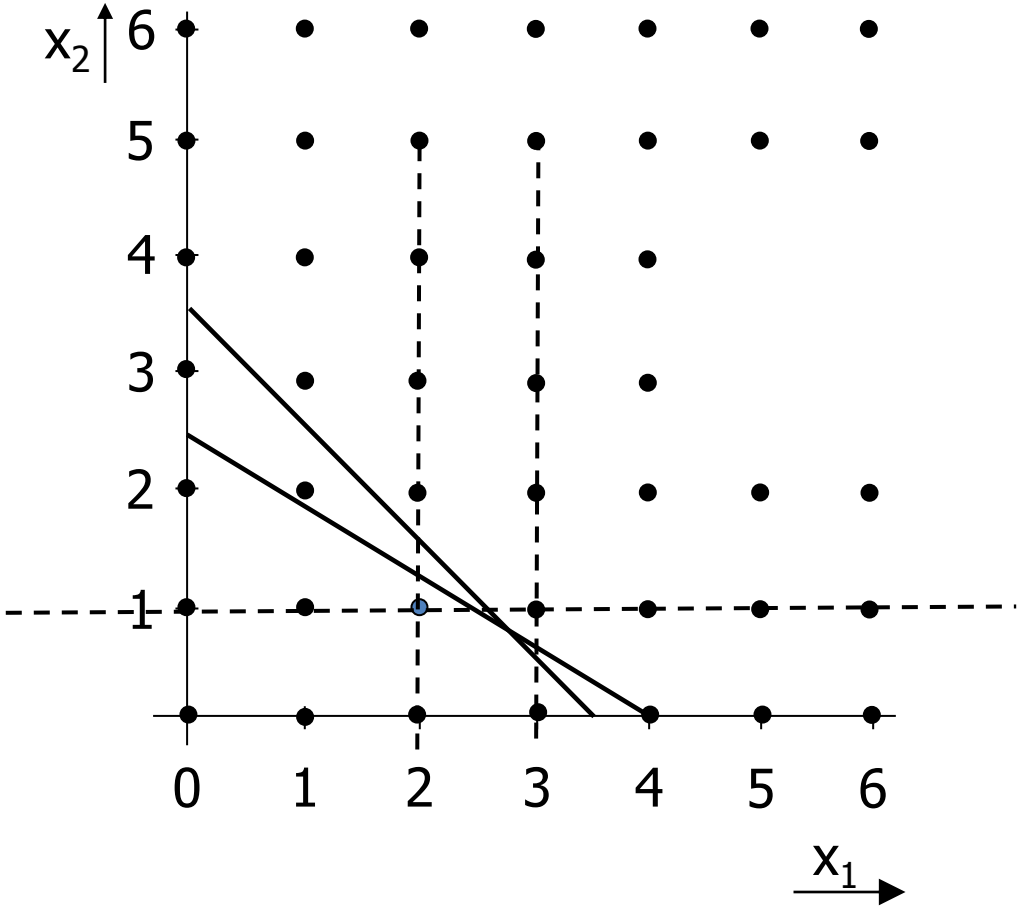


TRANSP-OR

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

Branch and Bound: Branching on x_2



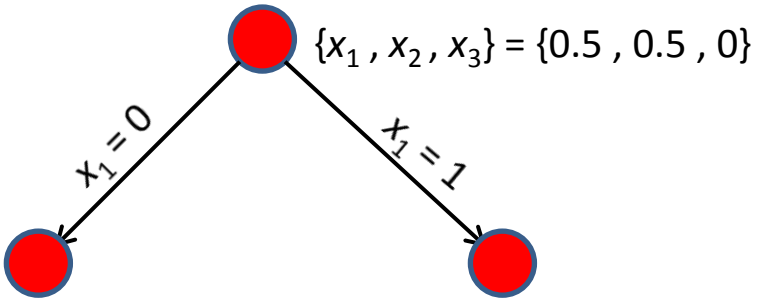


TRANSP-OR

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

Branch and Bound: Binary (0-1) Variable

- In both Fleet Assignment and Aircraft Routing, we formulated the model with binary variables. They can be solved by B&B technique as well
- Branching of a variable is done by fixing its value to 0 or 1



- How many LPs would a branch and bound algorithm solve?

Branch and Bound: Complexity

- Assuming that a branch and bound algorithm is used on a mathematical model with *m* binary variables. What is the *maximum* number of nodes (LPs) that need to be solved?
  - 1 variable $\Rightarrow$ 2 LPs
  - 2 variables $\Rightarrow$ 4 LPs
  - 10 variables $\Rightarrow$ 1024 LPs
  - *m* variables $\Rightarrow$ 2^m LPs
- For Fleet Assignment, note that 10 fleets and 1000 flights problem would have 10,000 binary variables (and more ground arc variables)
  - If we are *unfortunate*, optimal solution would be known after solving 2 x 10^{3010} LPs

Branch and Bound: Complexity

- However, there is good news!
- We need not always wait for eternity to solve so many LPs as we could
  - be pruning infeasible nodes
  - or, be pruning nodes as we already have a better bound
- For most practical problems, 99.99…% nodes get pruned on the path towards optimal solution
- But for a MIP with a few thousand variables, even 0.00…01% nodes could also mean a lot
  - But don’t worry, at every node, you have a LB and UB. You can stop the algorithm when optimality gap is reasonably small

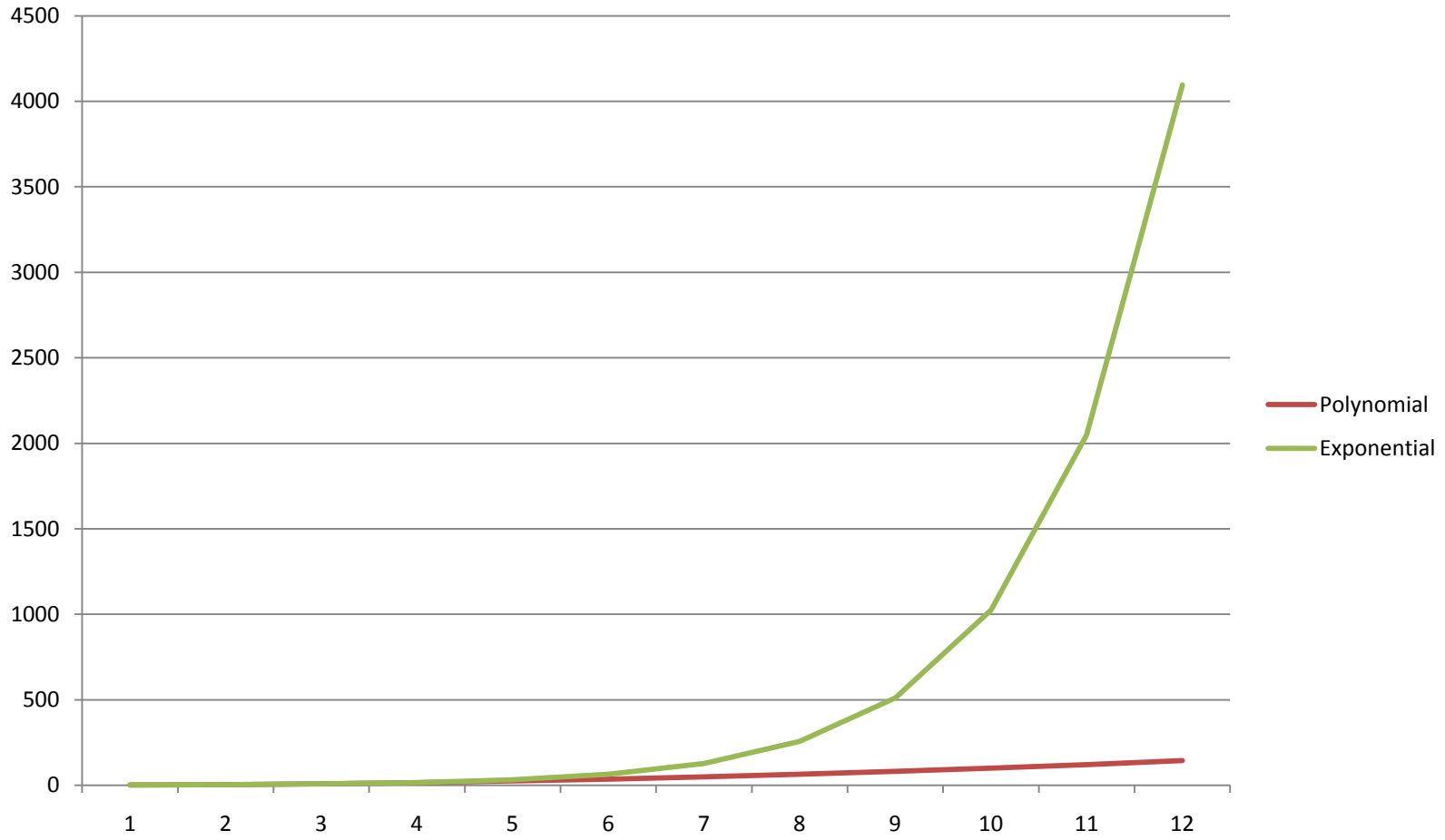
Algorithm Complexity

- Rewind back to the first session and meditate about the greedy algorithm and shortest path Dijkstra's algorithm
- Note that greedy algorithm takes the benefits and weights as inputs to determine value. These values are sorted and picked "greedily" to get the (optimal) solution.
  - Thus the complexity of the algorithm involved is the complexity involved in a mathematical operation, i.e. division, followed by sorting.
  - We know the best sorting algorithm's complexity is $n.\log n < n^2$ (n items)
- A graph with n nodes cannot have more than $n*(n-1)$ ($< n^2$) arcs. Since each arc is visited only once, the worst case complexity of Dijkstra's algorithm is bounded by n^2

Algorithm Complexity

- Thus, we can estimate the complexity of every algorithm based on variables (binary variables in assignment, arcs in a network etc.)
- Worst case complexity of the shortest path algorithm with n nodes is
  - 1 node $\Rightarrow$ 1 unit
  - 2 nodes $\Rightarrow$ 4 units
  - 10 nodes $\Rightarrow$ 100 units
  - 10,000 nodes $\Rightarrow$ 10^8 units
- Compare the incremental complexity of an algorithm that is bounded by polynomial (n^2) versus exponential (2^n)

Algorithm Complexity: P versus NP

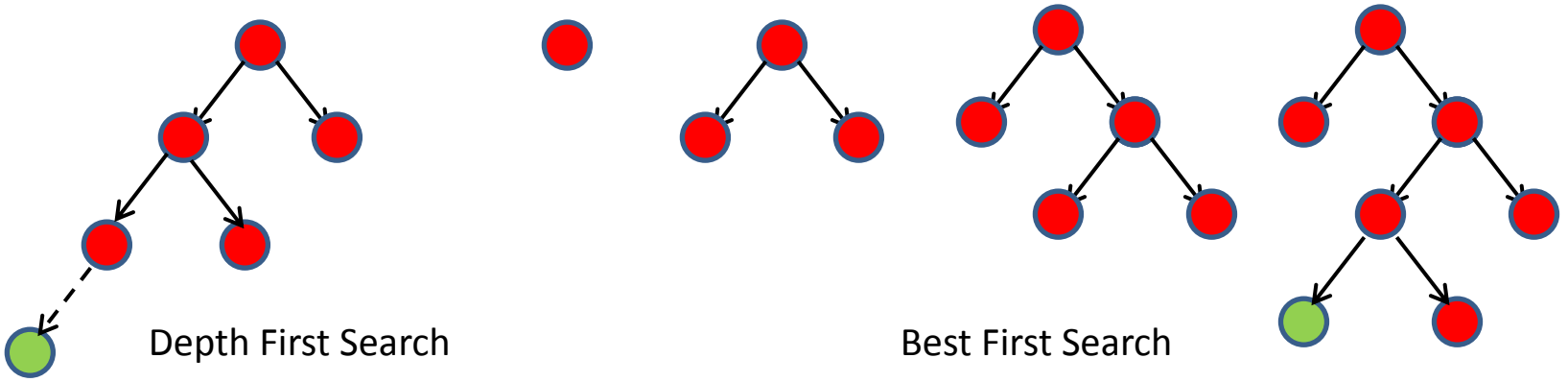


P versus NP

- Some problems are known to have polynomial time and space algorithms that produce optimal solutions – such as shortest path problem, transportation problem, minimal spanning tree etc.
  - These problems are categorized to be Polynomial (class P)
- Some other problems – such as travelling salesman problem, assignment problem, vehicle routing etc. – do not have any known polynomial time or space algorithms that can guarantee an optimal solution
  - These problems are categorized as Non-Polynomial (NP)
- Interestingly, the fact that there are no known NP algorithms for these problems does not mean that they cannot be P. It is just that there are no polynomial time algorithms that are known *so far*

Branch and Bound: Search Strategies

- Depth First Search: Deep dive on a single node to several child generations till you reach the last leaf node
- Best First Search: Expand the most "promising" node – for a minimization problem, it is node with the smallest UB and for a maximization problem, it is the node with the largest LB



- Which is better? Depth First or Best First?

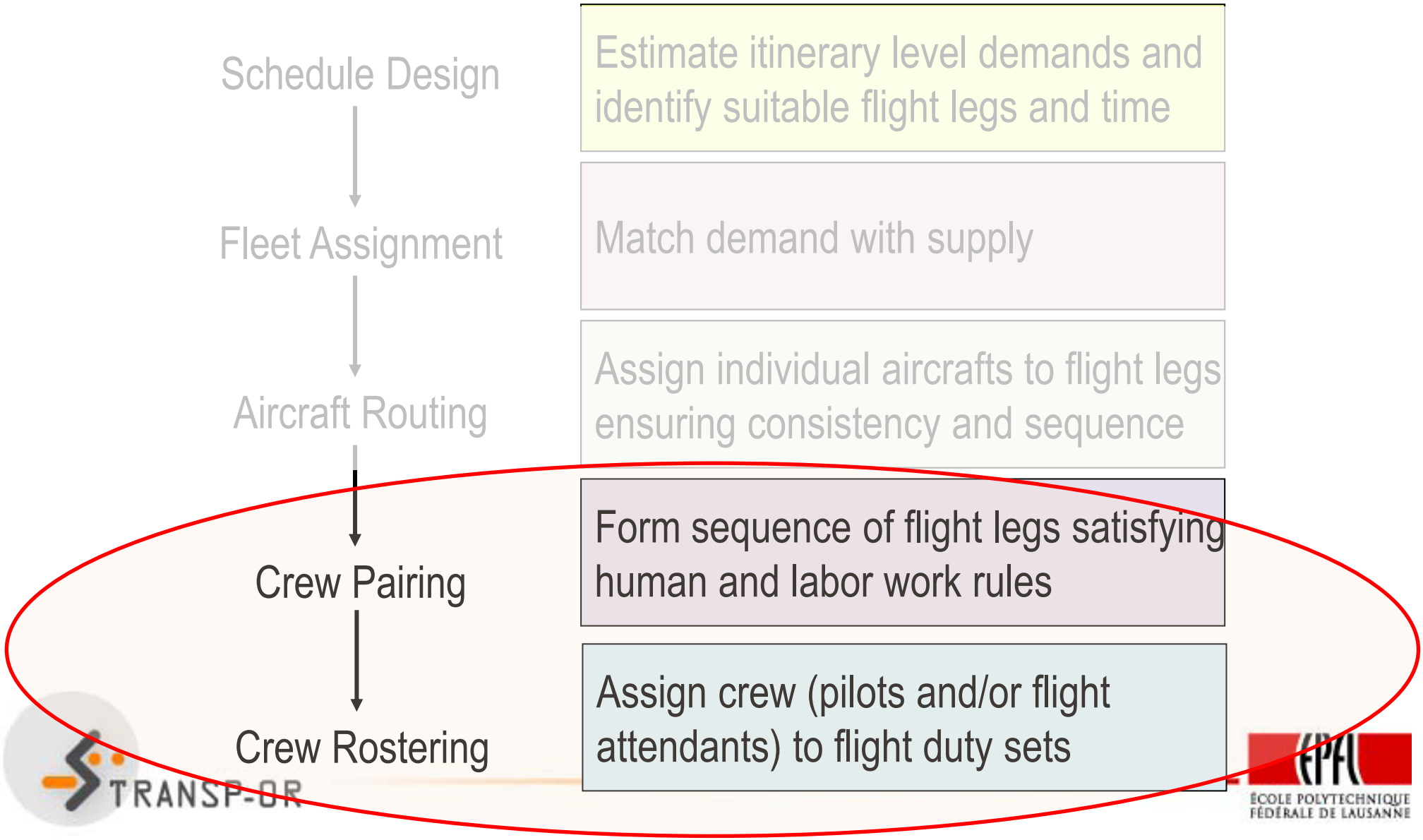
Branch and Bound: Commercial Solvers

- You can write your own routine (program) to solve an LP using simplex or interior point method
- This routine can be embedded in a search tree with different branching options to be solved as a branch and bound algorithm
- Alternatively you can use a commercial solver that have in-built branch and bound algorithm implemented
  - CPLEX
  - GLPK
  - Gurobi
  - …

Branch and Bound: Advancement

- The idea behind branch and bound algorithm is to reduce the feasible space by recursively removing the non-integral solutions from this space
- Note that this is done by adding new constraints (cuts) parallel to the axes
- But why should the cuts be only added parallel to the axes?
- This gives rise to the concept of cutting planes which are more efficient that cuts that are parallel to the axes
- The resulting algorithm is called **Branch and Cut**

Crew Scheduling



TRANSP-OR
ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

The Crew Scheduling Problem

- Assign crews to cover all flights for a given fleet type
- Minimize cost
  - Flying hours
  - Incidental expenses such as accommodation at non-domicile stations
  - Overtime and Deadheading
- Primary constraints
  - Cover constraint: A crew can be at most assigned to one flight at a time
  - Respect labor and regulatory laws relating to human work hours
- Side constraints
  - Balance: A crew starts and finishes her duty at a base / domicile station
  - Robustness

Some Terminologies

- Duty period:
  - A duty period is a day-long sequence of consecutive flights that can be assigned to a single crew, to be followed by a period of rest
- Duty rules:
  - Duty basically mimics a day's function of a crew. Thus a duty consists of a set of flights forming a sequence over space / time. Typical rules:

    Maximum flying time

    Minimum layover time

    Maximum layover time

    Maximum duty time

Some Terminologies

- A sequence of *duty periods*, interspersed with *periods of rest*, that begins and ends at a *crew domicile* (or *crew base*)
- A pairing usually extends over a period of 3-7 days and assumes the following rules:
  - First duty starts/last duty ends at the crew base
  - Duties are sequential in space/time
  - Minimum rest between duties
  - Number of overnights
  - Maximum number of days away from base
  - 8-in-24 rule

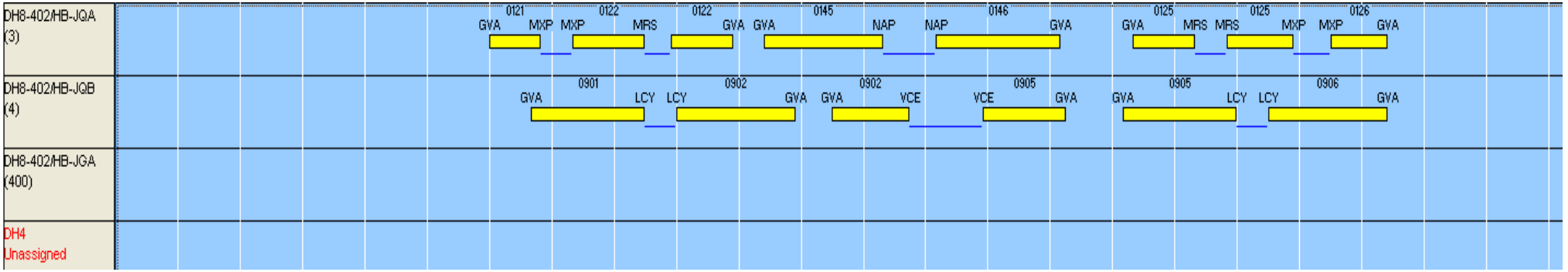
Cost Function

- Duty cost is computed as a maximum of:
  - Total flying time
  - f_d * total duty time
  - Minimum guaranteed duty pay
- Primarily compensates for flying time, but also compensates for "undesirable" schedules
- Pairing cost is computed as a maximum of:
  - Sum of duty costs
  - f_p * total time away from base (TAFB)
  - Minimum guaranteed pairing pay

Crew Pairing

- In smaller airline companies that does not operate any long haul flights, crew returns to the base after their duty period for the day
- In such situations, duty and pairing generation are one and the same problem. Even in larger airline companies, generation of duty is taken care of within the pairing generation process
- Objective of any crew pairing exercise is to minimize the number of pairings that can operate all flights in the schedule
- Problem is often decomposed as daily, weekly, exceptions or monthly
- **Deadheading** must be avoided to the extent possible

Crew Pairing: Arc-Node Representation

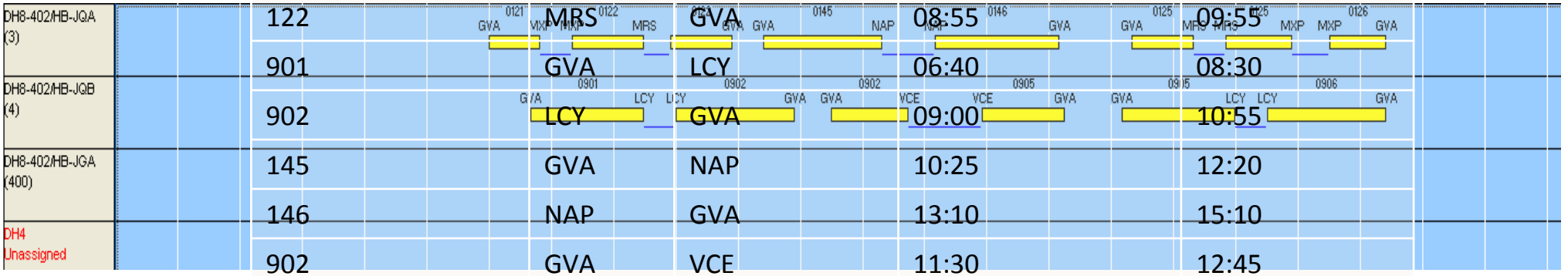


TRANSP-OR

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

Crew Pairing: Arc-Node Representation

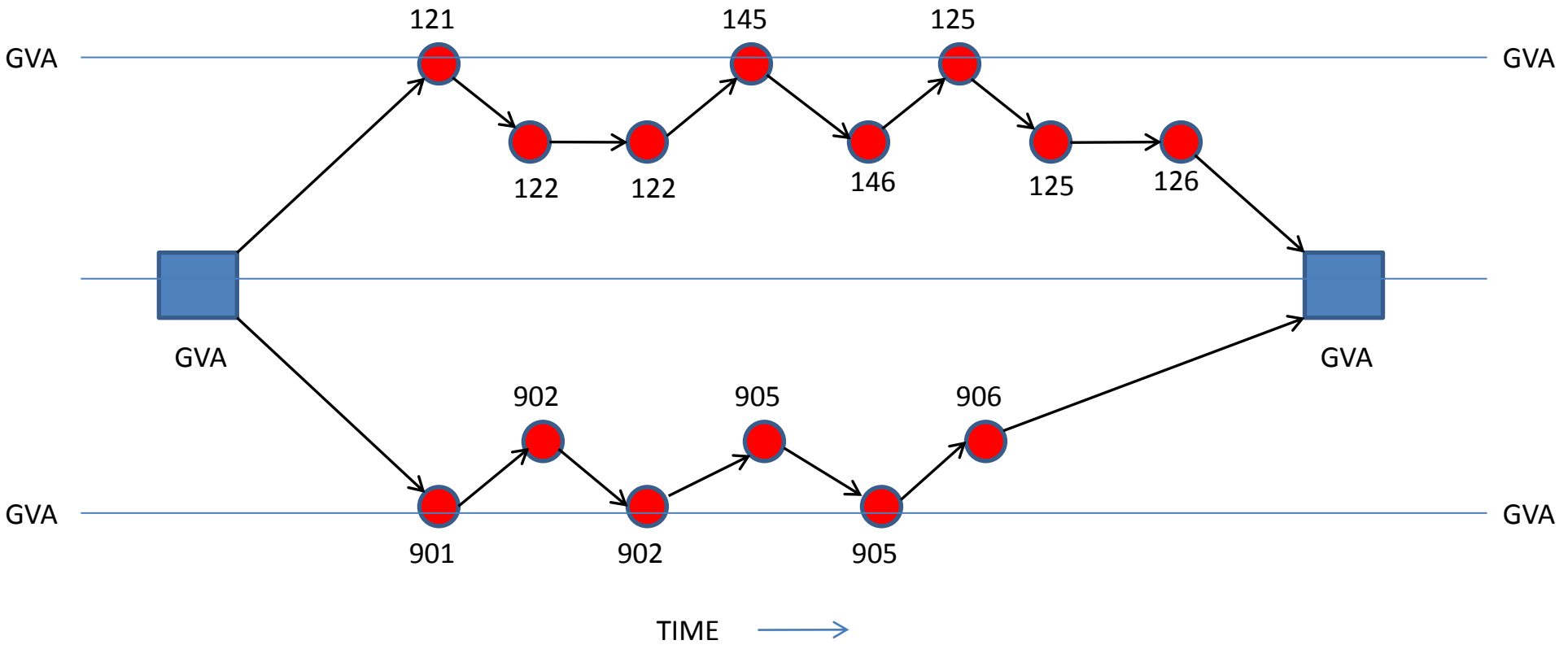
Flight Number	Origin	Destination	Departure Time	Arrival Time
121	GVA	MXP	06:00	06:50
122	MXP	MRS	07:20	08:30
122	MRS	GVA	08:55	09:55
901	GVA	LCY	06:40	08:30
902	LCY	GVA	09:00	10:55
145	GVA	NAP	10:25	12:20
146	NAP	GVA	13:10	15:10
902	GVA	VCE	11:30	12:45
905	VCE	GVA	13:55	15:15
905	GVA	LCY	16:10	18:00
906	LCY	GVA	18:30	20:25
125	GVA	MRS	16:20	17:20
125	MRS	MXP	17:50	18:55
126	MXP	GVA	19:30	20:25



TRANSP-OR

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

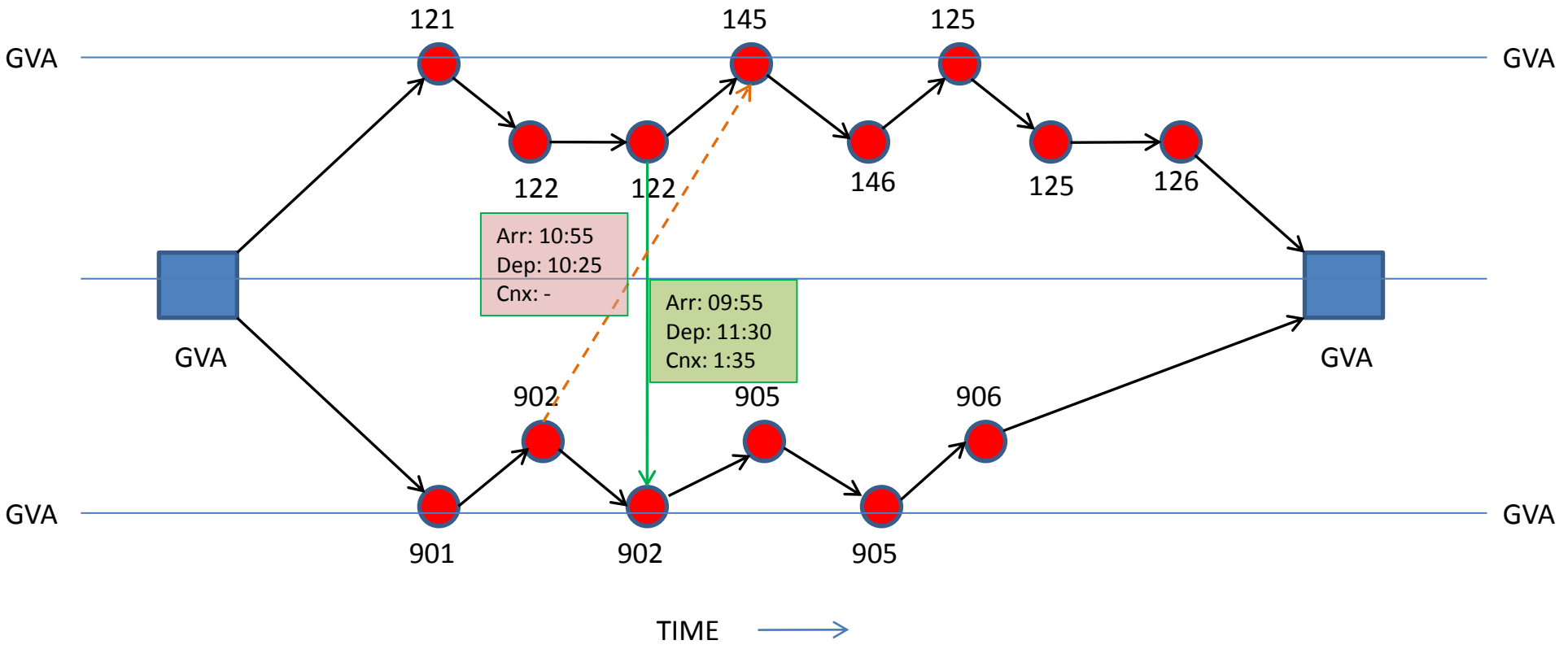
Crew Pairing: Node Representation of flights



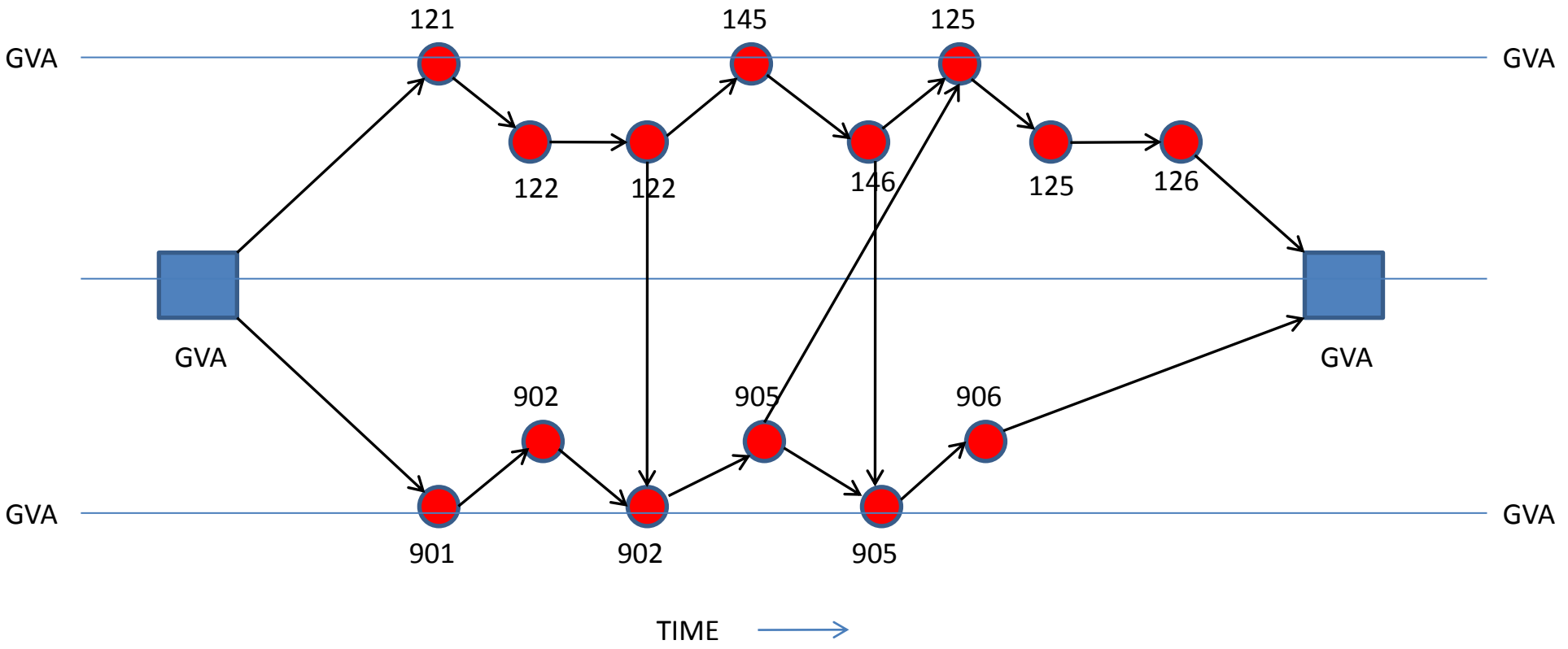
TRANSP-OR

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

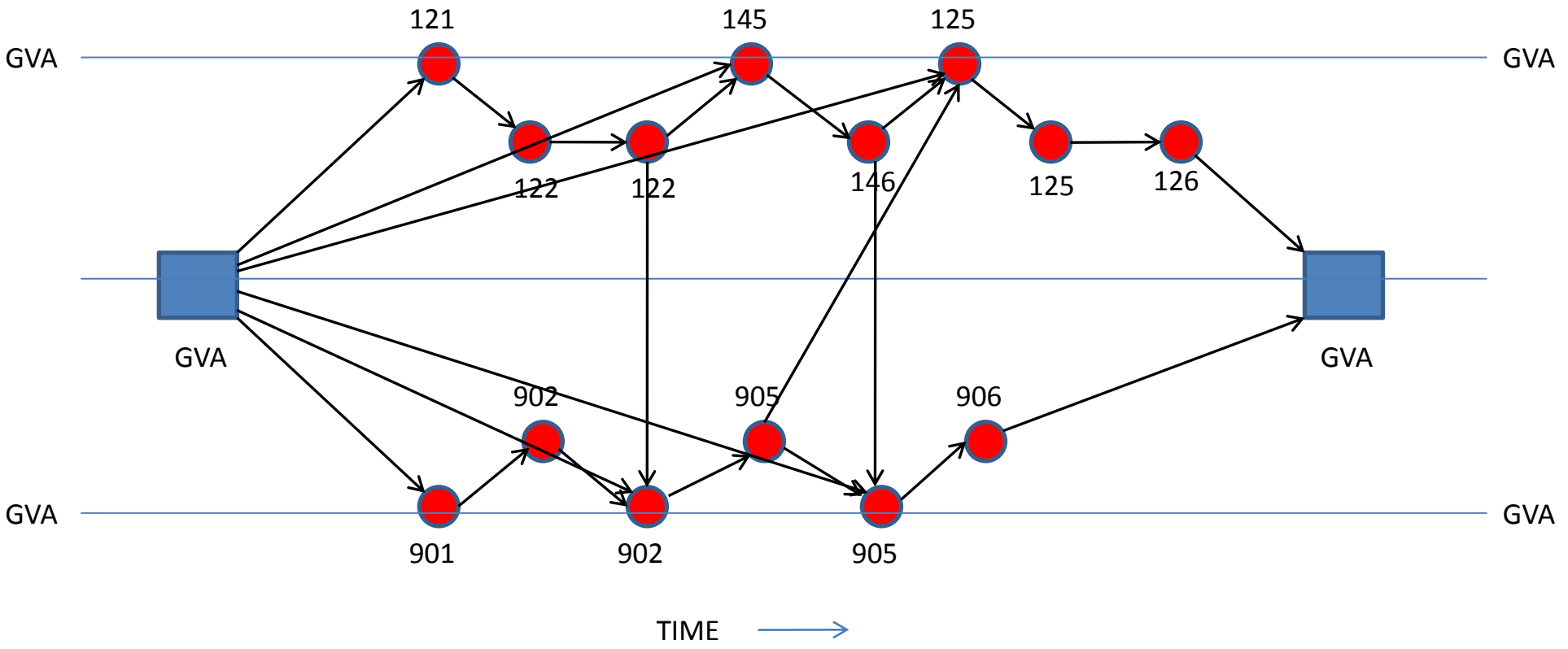
Crew Pairing: Node Representation of flights



Crew Pairing: Node Representation of flights



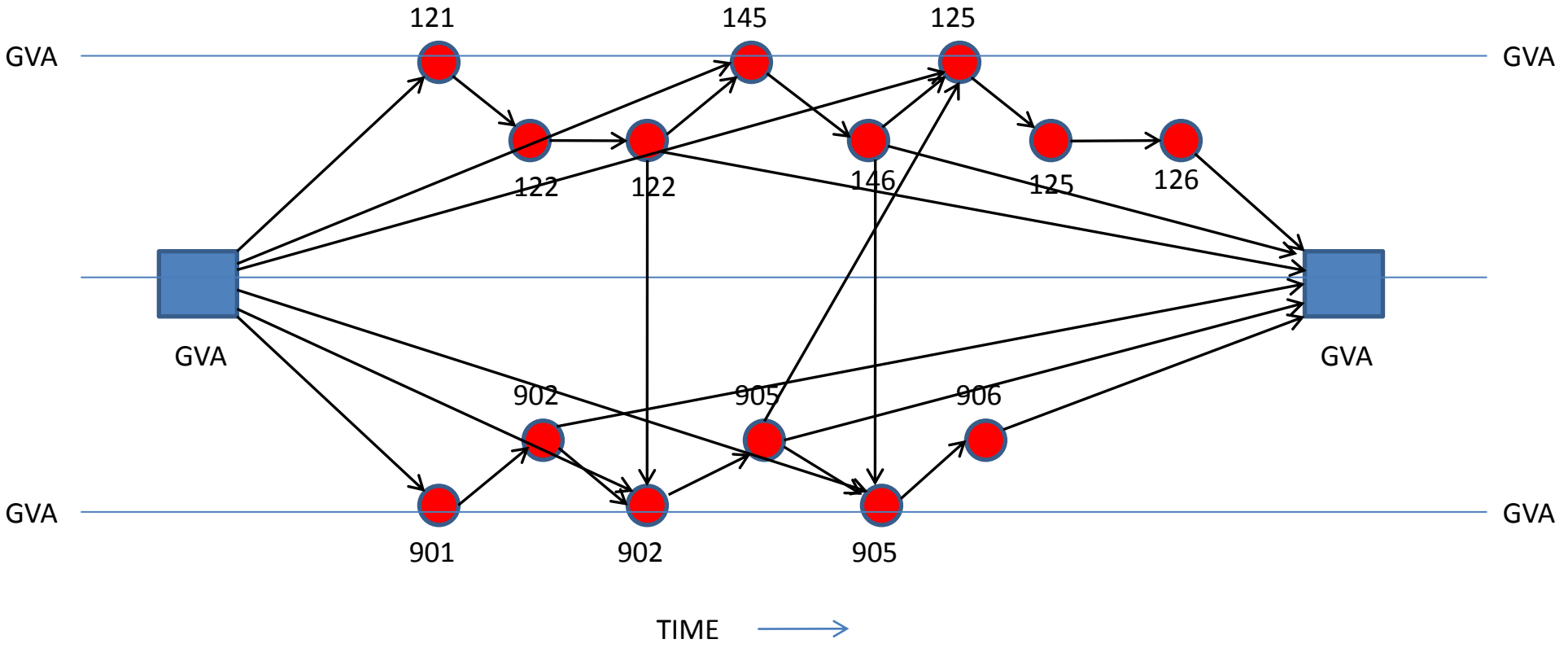
Crew Pairing: Node Representation of flights



TRANSP-OR

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

Crew Pairing: Node Representation of flights



TRANSP-OR

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

Crew Pairing: Problem Formulation

- The difference between duty and pairing blurs in the case of a smaller airline
  - As in this case, most crew members return to the base at the end of the day
- The number of feasible pairings increase drastically with the number of flights, even though the computation is pre-processed
- We have to find the minimal set of pairings that would cover all the flights – problem formulation is similar to aircraft routing
- If a flight is covered in more than one pairing, the crew would be deadheading – which must be avoided to the extent possible
- Note that the problem can be **decomposed** fleet-wise, airport (or base) station-wise, fleet-wise, crew type-wise etc

Crew Pairing Model: Notations

- Sets
  - Set of pairings for a specific fleet (P), indexed by p
  - Set of flights (F), indexed by f
- Parameters
  - c_p is the cost of pairing p
  - $A_{p,f}$ is 1 if pairing p has flight f, 0 otherwise
- Decision Variables
  - x_p equals 1 if pairing p is selected, and 0 otherwise

Crew Pairing Model: Formulation

$$\text{Minimize} \sum_{p \in P} c_p x_p$$

Subject to:

$$\sum_{p \in P} A_{p,f} x_p \geq 1, \ \forall f \in F$$

Bounds

$$x_p \in \{0,1\}$$

Crew Pairing Model: Solution Algorithm

- Of course, now that you are aware of B&B algorithm, you can use it solve this formulation
- However spare a thought: What would happen if there were several thousand flights for which crew needs to be assigned?
- Theoretically there could be several million (billion?) routes and even if only 1% of the routes are feasible, it would mean too many routes
- It might even be impossible to add all the routes to the model and compute

Crew Pairing Model: Column Generation

- Start with a subset of pairings to be input to the model
- This model with a subset of pairings variable is called the “relaxed” master problem (RMP)
- Generate new “interesting” pairings that are not already in the RMP
  - If there are such “interesting” pairings, add them to the RMP and resolve the new RMP
  - If there are NO such “interesting” pairings, RMP has all the pairings required in the optimal solution
- Start the B&B algorithm on this RMP
- This procedure is called *Branch-and-price*

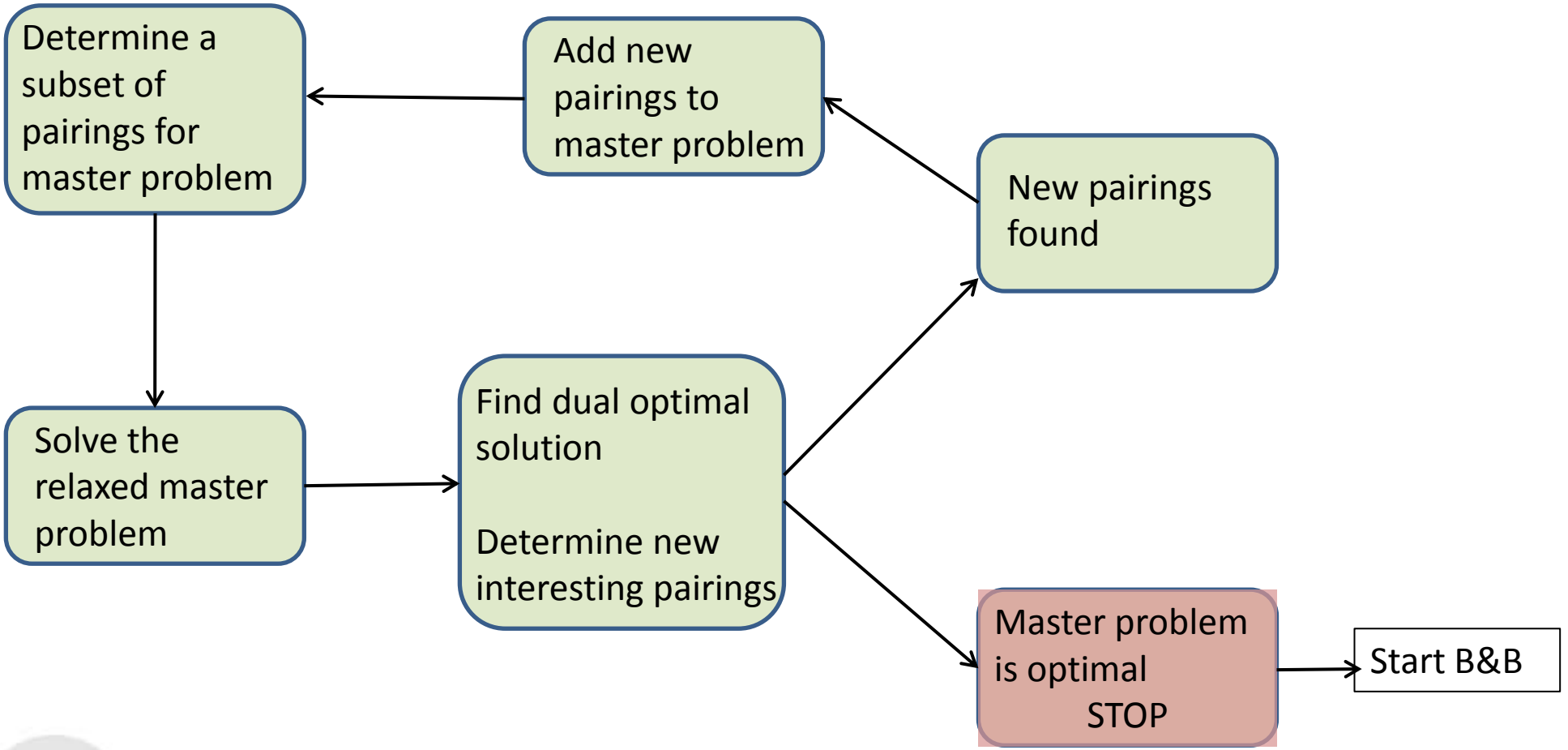
Crew Pairing: How to find “interesting” pairings?

- Start with a subset of columns such that you get a feasible solution
- These columns are part of the master problem. Solve the LP relaxation of the master problem and find the duals for all variables
- Since the primal variables represent the cost of the pairings, dual variables can be considered as the “profit” of flights
- Determine the reduced cost of pairings using $\bar{c} = c_p - \sum_{f \in F} A_{fp} \pi_f$
- Find one or more pairings with most negative reduced cost to be added to the master problem (how to find it?)
- Repeat this process till there are no pairings with negative reduced costs

Crew Pairing: Pricing as shortest path problem

- A pairing can be seen as a path, where nodes represent flights and arcs represent valid connections
- Some of the feasibility conditions imposed on a path to decompose the problem are as follows:
  - Paths must start and end at a given crew domicile. For example, if all crew members of Baboo are based in Geneva, no path can start or end at a station other than GVA
  - Paths cannot repeat the same flight for the same day
  - Paths must satisfy duty and pairing rules (remember rules such as 8-in-24 etc.)
  - Path costs can be computed using the labels corresponding to pairing reduced costs

Crew Pairing: Column Generation Summary



Column Generation Algorithm Review

- What happens if the algorithm does not find the set of optimal columns even after several thousands of iterations?
  - Of course, you can STOP the optimizer and choose the best result so far
- Note that column generation algorithm can be applied to aircraft maintenance routing problem as well
  - Algorithm can be applied to all “set covering” formulations
  - Incidentally, column generation algorithms can be applied to all class of problems with lot more variables (columns) than constraints (rows)

Crew Rostering

- Crew rostering is the process of assigning actual crew members – Flight Commanders, First Officers and Cabin Crew – to different pairings
- Input
  - A set of pairings (output from crew pairing)
  - A set of crew members with required qualifications
  - Crew availability and preferences
- Output
  - An assignment of crews to pairings

Crew Rostering

Typical Crew Rostering Rules:

- Minimum rest between consecutive pairings
- Maximum flying time over a month
- Vacations and Training requirements for crew members
- Minimum total number of days off

Main difference:

- Focus on crew preferences rather than profitability of the schedule

The Bidline Problem

- Anonymous Pairings are constructed at the outset
- These pairings are combined over a longer time horizon, usually a month, to form a schedule
- Schedules are posted and crew members bid for specific schedules
- Senior crew members given higher priority
- Commonly used in the U.S.

Individualized Rostering

- Crew vacation requests, training needs, etc. are taken as inputs
- Monthly schedules are generated based on specific requests
- Planners accept or reject the requests
- Priority of requests based on seniority
- Typical used in Europe and Asia

Crew Rostering: Solution Methodology

- Pairings are combined to form schedules – much the same way as duties or individual flights are combined to form the pairings
- Problem usually solved using a branch-and-price algorithm such that every pairing is assigned to a crew member of each type
- In Europe, pairings for each individual crew member is formed based on the wishes
- Problem is solved jointly for all crew members such that all pairings are assigned and the cost is kept at the minimum

Pairing versus Rostering

- Similarities
  - Sequencing flights or duties to form pairings while sequencing pairings to form schedules (rosters)
  - Set partitioning formulations solved using B&P algorithms
- Differences
  - Anonymous schedules versus personalised schedules for each crew member
  - Time horizon of pairing is shorter than time horizon for rostering
- Integration
  - Ideally both the problems should be solved jointly, but the problem size grows up
  - Recent research aims to combine pairing and rostering

Cockpit versus Cabin Crew

- Cockpit crew often impose requests to fly as a specific team – particular First Officer – sometimes a particular cabin crew
- Most of the cockpit crew members vary only by the fleet type, however cabin crew might require specific qualifications (example, language skills)
- Cabin crew members have a wider range of fleet types they can staff
- What do you think happens in very long haul flights – example flights between Chicago to Sydney or Narita?

Reserve Crew Members

- Crew members, especially cockpit crew members, are given flexibility to decide about flying on their own
- They may refuse to fly if they feel physically or mentally unfit – No need to undergo any medical tests
- Thus there are possibilities for absenteeism
- This is usually taken care of by providing adequate “reserve” crews (on-call) at major crew domiciles
- What would be the optimal number of reserve crews at a specific airport?

Robust Scheduling and Recovery

- Hot topic of research
- Given a disruption or delay, find pairings and rosters such that it would have maximum opportunities to “swap” around
- Basic objectives of a robust, recoverable schedule are:
  - Return to original schedule quickly and with least disruption
  - Minimize passenger disruptions
- Limited time horizon -- need fast heuristics