Discrete Optimization

Quentin Louveaux

ULg - Institut Montefiore

2015

Discrete

What is discreteness ?

Anything that should be considered as a whole and cannot be divided.

Life is full of discrete decisions !

- The clothes to wear today
- A route to select to come to the campus
- A course to select
- The food you eat for lunch
- whether you watch TV or you go out
- ...

Discrete

What is discreteness ?

Anything that should be considered as a whole and cannot be divided.

Life is full of discrete decisions !

- The clothes to wear today
- A route to select to come to the campus
- A course to select
- The food you eat for lunch
- whether you watch TV or you go out
- ...

Discrete

What is discreteness ?

Anything that should be considered as a whole and cannot be divided.

Life is full of discrete decisions !

- The clothes to wear today
- A route to select to come to the campus
- A course to select
- The food you eat for lunch
- whether you watch TV or you go out
- ...

Discrete

What is discreteness ?

Anything that should be considered as a whole and cannot be divided.

Life is full of discrete decisions !

- The clothes to wear today
- A route to select to come to the campus
- A course to select
- The food you eat for lunch
- whether you watch TV or you go out
- ...

Discrete

What is discreteness ?

Anything that should be considered as a whole and cannot be divided.

Life is full of discrete decisions !

- The clothes to wear today
- A route to select to come to the campus
- A course to select
- The food you eat for lunch
- whether you watch TV or you go out
- ...

Discrete

What is discreteness ?

Anything that should be considered as a whole and cannot be divided.

Life is full of discrete decisions !

- The clothes to wear today
- A route to select to come to the campus
- A course to select
- The food you eat for lunch
- whether you watch TV or you go out
- ...

Discrete

What is discreteness ?

Anything that should be considered as a whole and cannot be divided.

Life is full of discrete decisions !

- The clothes to wear today
- A route to select to come to the campus
- A course to select
- The food you eat for lunch
- whether you watch TV or you go out
- . . .

Discrete

What is discreteness ?

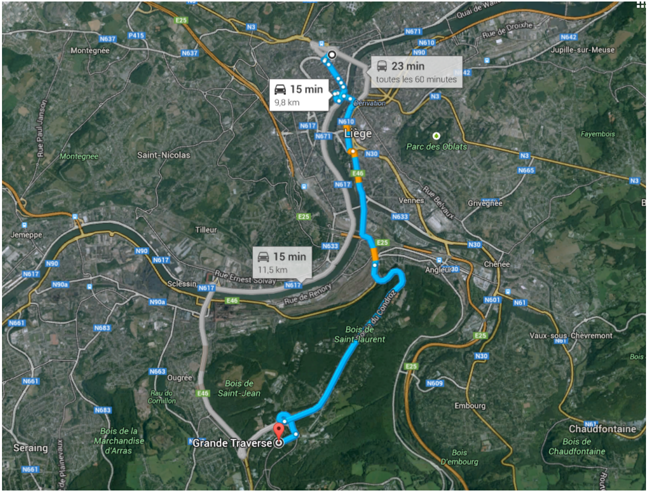
Anything that should be considered as a whole and cannot be divided.

Life is full of discrete decisions !

- The clothes to wear today
- A route to select to come to the campus
- A course to select
- The food you eat for lunch
- whether you watch TV or you go out
- . . .

Some well-known discrete problems

The shortest path

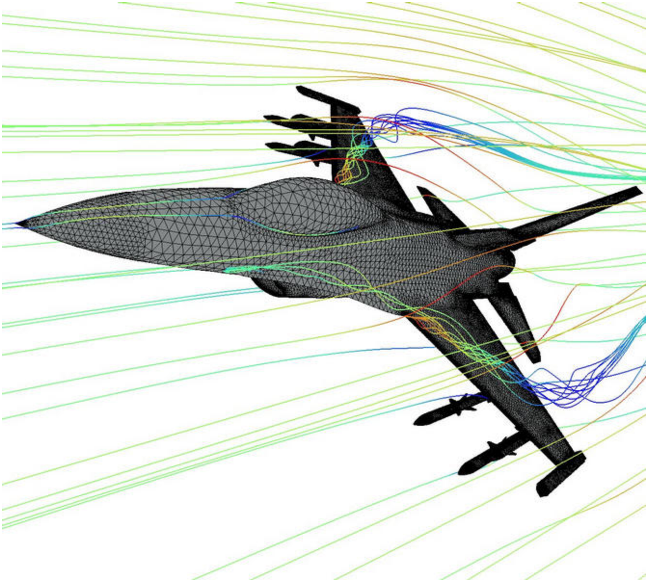


Some well-known discrete problems

The sorting problem

A lesser known but tractable problem

The matching problem



A popular discrete problem : the traveling salesman problem

Probably the most well-known discrete problem. . .

With many applications : Vehicle routing, VLSI design,. . .

Given n cities, in which order should one visit them in order to minimize the total distance ?

Example : Visit 23 EU cities with minimal traveling distance

Optimal solution

A popular discrete problem : the traveling salesman problem

Probably the most well-known discrete problem. . .
With many applications : Vehicle routing, VLSI design,. . .

Given n cities, in which order should one visit them in order to minimize the total distance ?

Example : Visit 23 EU cities with minimal traveling distance
Optimal solution

A popular discrete problem : the traveling salesman problem

Probably the most well-known discrete problem...
With many applications : Vehicle routing, VLSI design,...

Given n cities, in which order should one visit them in order to minimize the total distance ?

Example : Visit 23 EU cities with minimal traveling distance

Optimal solution

A popular discrete problem : the traveling salesman problem

Probably the most well-known discrete problem...
With many applications : Vehicle routing, VLSI design,...

Given n cities, in which order should one visit them in order to minimize the total distance ?

Example : Visit 23 EU cities with minimal traveling distance

Optimal solution

Why is that so complicated ?

After all, there is a finite number of solutions
In particular $n!$ possible permutations

Imagine we can check 10^{12} possibilities per second
That is already a pretty amazing machine...

10 !	0 sec
20 !	28 days
30 !	8400 billion years
40 !	5 quadrillions times the age of the Earth...

Why is that so complicated ?

After all, there is a finite number of solutions
In particular $n!$ possible permutations

Imagine we can check 10^{12} possibilities per second
That is already a pretty amazing machine. . .

10 !	0 sec
20 !	28 days
30 !	8400 billion years
40 !	5 quadrillions times the age of the Earth. . .

Why is that so complicated ?

After all, there is a finite number of solutions
In particular $n!$ possible permutations

Imagine we can check 10^{12} possibilities per second
That is already a pretty amazing machine. . .

10 !	0 sec
20 !	28 days
30 !	8400 billion years
40 !	5 quadrillions times the age of the Earth. . .

Pure enumeration is hopeless

An algorithm based on exponential running time has an inherent limit.
Example : Enumerate all 2^n potential solutions of a problem with n binary choices.

n	Computer 1	Computer 2
Checks	10^9 per second	10^{15} per second
10	0 sec	0 sec
20	0 sec	0 sec
30	1 sec	0 sec
40	18 min	0 sec
50	13 days	1 sec
60	36 years	19 min
70	37000 y.	13 days
80	38 million years	38 years

To solve $n \approx 500$, one would need a computer 10^{140} times faster.
It is however possible to deal with relatively small instances.
Today we can deal routinely with problems with 1000 discrete variables

Pure enumeration is hopeless

An algorithm based on exponential running time has an inherent limit.
Example : Enumerate all 2^n potential solutions of a problem with n binary choices.

n	Computer 1	Computer 2
Checks	10^9 per second	10^{15} per second
10	0 sec	0 sec
20	0 sec	0 sec
30	1 sec	0 sec
40	18 min	0 sec
50	13 days	1 sec
60	36 years	19 min
70	37000 y.	13 days
80	38 million years	38 years

To solve $n \approx 500$, one would need a computer 10^{140} times faster.
It is however possible to deal with relatively small instances.
Today we can deal routinely with problems with 1000 discrete variables

Pure enumeration is hopeless

An algorithm based on exponential running time has an inherent limit.
Example : Enumerate all 2^n potential solutions of a problem with n binary choices.

n	Computer 1	Computer 2
Checks	10^9 per second	10^{15} per second
10	0 sec	0 sec
20	0 sec	0 sec
30	1 sec	0 sec
40	18 min	0 sec
50	13 days	1 sec
60	36 years	19 min
70	37000 y.	13 days
80	38 million years	38 years

To solve $n \approx 500$, one would need a computer 10^{140} times faster.
It is however possible to deal with relatively small instances.
Today we can deal routinely with problems with 1000 discrete variables

Pure enumeration is hopeless

An algorithm based on exponential running time has an inherent limit.
Example : Enumerate all 2^n potential solutions of a problem with n binary choices.

n	Computer 1	Computer 2
Checks	10^9 per second	10^{15} per second
10	0 sec	0 sec
20	0 sec	0 sec
30	1 sec	0 sec
40	18 min	0 sec
50	13 days	1 sec
60	36 years	19 min
70	37000 y.	13 days
80	38 million years	38 years

To solve $n \approx 500$, one would need a computer 10^{140} times faster.
It is however possible to deal with relatively small instances.
Today we can deal routinely with problems with 1000 discrete variables

Pure enumeration is hopeless

An algorithm based on exponential running time has an inherent limit.
Example : Enumerate all 2^n potential solutions of a problem with n binary choices.

n	Computer 1	Computer 2
Checks	10^9 per second	10^{15} per second
10	0 sec	0 sec
20	0 sec	0 sec
30	1 sec	0 sec
40	18 min	0 sec
50	13 days	1 sec
60	36 years	19 min
70	37000 y.	13 days
80	38 million years	38 years

To solve $n \approx 500$, one would need a computer 10^{140} times faster.
It is however possible to deal with relatively small instances.
Today we can deal routinely with problems with 1000 discrete variables

Applications of the TSP

- Truck routing
- Arranging school buses routes (the very first application)
- Scheduling of a machine to drill holes in a circuit board
- Delivery of meals to homebound people
- Genome sequencing
  Cities are local strings, and the cost is the measure of likelihood that a sequence follows another
- Link points through fiber optic connections in order to minimize the total distance and ensure that any failure leaves the whole system operational

Milestones in the solution of TSP instances

Year	Research team	Number of cities
1954	Dantzig, Fulkerson, Johnson	49 cities
1971	Held, Karp	64 cities
1977	Grötschel	120 cities
1980	Crowder, Padeberg	318 cities
1987	Padberg, Rinaldi	2392 cities
1994	Applegate, Bixby, Chvatal, Cook	7397 cities
2006	Applegate, Bixby, Chvatal, Cook, Helsgaun	80 000 cities

There is a $ 1000 prize for a challenge of 100 000 cities

Techniques : formulating as an integer programming problem, branch-and-bound, cutting planes.
→ topics covered in the class

Milestones in the solution of TSP instances

Year	Research team	Number of cities
1954	Dantzig, Fulkerson, Johnson	49 cities
1971	Held, Karp	64 cities
1977	Grötschel	120 cities
1980	Crowder, Padeberg	318 cities
1987	Padberg, Rinaldi	2392 cities
1994	Applegate, Bixby, Chvatal, Cook	7397 cities
2006	Applegate, Bixby, Chvatal, Cook, Helsgaun	80 000 cities

There is a $ 1000 prize for a challenge of 100 000 cities

Techniques : formulating as an integer programming problem, branch-and-bound, cutting planes.
→ topics covered in the class

Milestones in the solution of TSP instances

Year	Research team	Number of cities
1954	Dantzig, Fulkerson, Johnson	49 cities
1971	Held, Karp	64 cities
1977	Grötschel	120 cities
1980	Crowder, Padeberg	318 cities
1987	Padberg, Rinaldi	2392 cities
1994	Applegate, Bixby, Chvatal, Cook	7397 cities
2006	Applegate, Bixby, Chvatal, Cook, Helsgaun	80 000 cities

There is a \$ 1000 prize for a challenge of 100 000 cities

Techniques : formulating as an integer programming problem, branch-and-bound, cutting planes.
$\rightarrow$ topics covered in the class

Learning outcomes

- Being able to model discrete problems
- Being able to recognize a good from a bad formulation
- Being able to recognize well-solved discrete problems from NP-hard ones
- Know the main algorithmic techniques to solve the easy and hard discrete problems

Learning outcomes

- Being able to model discrete problems
- Being able to recognize a good from a bad formulation
- Being able to recognize well-solved discrete problems from NP-hard ones
- Know the main algorithmic techniques to solve the easy and hard discrete problems

Learning outcomes

- Being able to model discrete problems
- Being able to recognize a good from a bad formulation
- Being able to recognize well-solved discrete problems from NP-hard ones
- Know the main algorithmic techniques to solve the easy and hard discrete problems

Learning outcomes

- Being able to model discrete problems
- Being able to recognize a good from a bad formulation
- Being able to recognize well-solved discrete problems from NP-hard ones
- Know the main algorithmic techniques to solve the easy and hard discrete problems

Organization

Schedule

- Lecture at 2 :00 pm every Friday
- Exercises follow the lecture (but not today)
- 2 Modeling and Implementation projects (one individual and by groups of 2)
  Will start very soon !

Grading

- The two projects count for 1/2 of the mark.
- An exercise exam (**written**) counts for 1/2 of the mark.
- I use a geometric mean, i.e.

$$Grade = \sqrt{Exam \times \left(\frac{Project\ 1 + Project\ 2}{2}\right)}$$

Organization

Schedule

- Lecture at 2 :00 pm every Friday
- Exercises follow the lecture (but not today)
- 2 Modeling and Implementation projects (one individual and by groups of 2)
  Will start very soon !

Grading

- The two projects count for 1/2 of the mark.
- An exercise exam (**written**) counts for 1/2 of the mark.
- I use a geometric mean, i.e.

$$Grade = \sqrt{Exam \times (\frac{Project\ 1 + Project\ 2}{2})}$$

Modeling

The main focus of the lecture is to formulate problems using

mathematical optimization formulations.

That means that we want to define variables, mathematical constraints (inequalities and equalities essentially) and an objective function to maximize or minimize.

$$\begin{aligned} \min\ & c(x) \\ \text{s.t.}\ & f(x) \le b \\ & g(x) = 0 \\ & x \in X. \end{aligned}$$

Why mathematical programming ?

Typical programming approach

- Analyze the problem
- Write an algorithm to solve the problem using `while`, `if`, `then`, `else`,...
- Proof the corectness of the algorithm
- Analyze the complexity of the algorithm

Pros and cons

- Works well for well-posed problems.
- Works well for tractable problems.
- Needs to be done from scratch if there is a slight change in the problem or additional constraints

Why mathematical programming ?

Typical programming approach

- Analyze the problem
- Write an algorithm to solve the problem using `while`, `if`, `then`, `else`,...
- Proof the corectness of the algorithm
- Analyze the complexity of the algorithm

Pros and cons

- Works well for well-posed problems.
- Works well for tractable problems.
- Needs to be done from scratch if there is a slight change in the problem or additional constraints

Why mathematical programming?

Typical programming approach

- Analyze the problem
- Write an algorithm to solve the problem using `while`, `if`, `then`, `else`,...
- Proof the corectness of the algorithm
- Analyze the complexity of the algorithm

Pros and cons

- Works well for well-posed problems.
- Works well for tractable problems.
- Needs to be done from scratch if there is a slight change in the problem or additional constraints

Why mathematical programming ?

Typical programming approach

- Analyze the problem
- Write an algorithm to solve the problem using `while`, `if`, `then`, `else`,. . .
- Proof the corectness of the algorithm
- Analyze the complexity of the algorithm

Pros and cons

- Works well for well-posed problems.
- Works well for tractable problems.
- Needs to be done from scratch if there is a slight change in the problem or additional constraints

Why mathematical programming ?

Typical programming approach

- Analyze the problem
- Write an algorithm to solve the problem using `while`, `if`, `then`, `else`,. . .
- Proof the corectness of the algorithm
- Analyze the complexity of the algorithm

Pros and cons

- Works well for well-posed problems.
- Works well for tractable problems.
- Needs to be done from scratch if there is a slight change in the problem or additional constraints

Why mathematical programming ?

Typical programming approach

- Analyze the problem
- Write an algorithm to solve the problem using `while`, `if`, `then`, `else`,...
- Proof the corectness of the algorithm
- Analyze the complexity of the algorithm

Pros and cons

- Works well for well-posed problems.
- Works well for tractable problems.
- Needs to be done from scratch if there is a slight change in the problem or additional constraints

Why mathematical programming ?

Typical programming approach

- Analyze the problem
- Write an algorithm to solve the problem using `while`, `if`, `then`, `else`,...
- Proof the corectness of the algorithm
- Analyze the complexity of the algorithm

Pros and cons

- Works well for well-posed problems.
- Works well for tractable problems.
- Needs to be done from scratch if there is a slight change in the problem or additional constraints

Why mathematical programming ?

Typical programming approach

- Analyze the problem
- Write an algorithm to solve the problem using `while`, `if`, `then`, `else`,...
- Proof the corectness of the algorithm
- Analyze the complexity of the algorithm

Pros and cons

- Works well for well-posed problems.
- Works well for tractable problems.
- Needs to be done from scratch if there is a slight change in the problem or additional constraints

Why mathematical programming ?

Typical programming approach

- Analyze the problem
- Write an algorithm to solve the problem using `while`, `if`, `then`, `else`,. . .
- Proof the corectness of the algorithm
- Analyze the complexity of the algorithm

Pros and cons

- Works well for well-posed problems.
- Works well for tractable problems.
- Needs to be done from scratch if there is a slight change in the problem or additional constraints

Why mathematical programming?

The mathematical programming approach

- Analyze the problem
- Write a static mathematical model
- Rely on meta-algorithms that work on all models that are correctly written

Pros and cons

- Sometimes difficult to formulate the problems in the right way (no if, then, else)
- Works also for hard problems
- Two different models of the same problem may not be as good as the other
- Very flexible to add complicating constraints.

Why mathematical programming ?

The mathematical programming approach

- Analyze the problem
- Write a static mathematical model
- Rely on meta-algorithms that work on all models that are correctly written

Pros and cons

- Sometimes difficult to formulate the problems in the right way (no `if`, `then`, `else`)
- Works also for hard problems
- Two different models of the same problem may not be as good as the other
- Very flexible to add complicating constraints.

Why mathematical programming?

The mathematical programming approach

- Analyze the problem
- Write a static mathematical model
- Rely on meta-algorithms that work on all models that are correctly written

Pros and cons

- Sometimes difficult to formulate the problems in the right way (no `if`, `then`, `else`)
- Works also for hard problems
- Two different models of the same problem may not be as good as the other
- Very flexible to add complicating constraints.

Why mathematical programming ?

The mathematical programming approach

- Analyze the problem
- Write a static mathematical model
- Rely on meta-algorithms that work on all models that are correctly written

Pros and cons

- Sometimes difficult to formulate the problems in the right way (no `if`, `then`, `else`)
- Works also for hard problems
- Two different models of the same problem may not be as good as the other
- Very flexible to add complicating constraints.

Why mathematical programming ?

The mathematical programming approach

- Analyze the problem
- Write a static mathematical model
- Rely on meta-algorithms that work on all models that are correctly written

Pros and cons

- Sometimes difficult to formulate the problems in the right way (no `if`, `then`, `else`)
- Works also for hard problems
- Two different models of the same problem may not be as good as the other
- Very flexible to add complicating constraints.

Why mathematical programming ?

The mathematical programming approach

- Analyze the problem
- Write a static mathematical model
- Rely on meta-algorithms that work on all models that are correctly written

Pros and cons

- Sometimes difficult to formulate the problems in the right way (no `if`, `then`, `else`)
- Works also for hard problems
- Two different models of the same problem may not be as good as the other
- Very flexible to add complicating constraints.

Why mathematical programming ?

The mathematical programming approach

- Analyze the problem
- Write a static mathematical model
- Rely on meta-algorithms that work on all models that are correctly written

Pros and cons

- Sometimes difficult to formulate the problems in the right way (no `if`, `then`, `else`)
- Works also for hard problems
- Two different models of the same problem may not be as good as the other
- Very flexible to add complicating constraints.

Why mathematical programming ?

The mathematical programming approach

- Analyze the problem
- Write a static mathematical model
- Rely on meta-algorithms that work on all models that are correctly written

Pros and cons

- Sometimes difficult to formulate the problems in the right way (no `if`, `then`, `else`)
- Works also for hard problems
- Two different models of the same problem may not be as good as the other
- Very flexible to add complicating constraints.

Why mathematical programming ?

The mathematical programming approach

- Analyze the problem
- Write a static mathematical model
- Rely on meta-algorithms that work on all models that are correctly written

Pros and cons

- Sometimes difficult to formulate the problems in the right way (no `if`, `then`, `else`)
- Works also for hard problems
- Two different models of the same problem may not be as good as the other
- Very flexible to add complicating constraints.

Some random applications

Location of GSM transmitters

A mobile phone operator decides to equip a currently uncovered geographical zone.

The management allocates a budget of 10 million € to equip this region.

7 locations are possible for the construction of the transmitters.

Every transmitter only covers a certain number of communities.

Where should the transmitters be built to cover the largest population with the given budget ?

Electricity production

An electricity producer wants to plan the level of production of his main power plants in order to fulfill the demand and minimize his costs. The demand for 6 time periods in the day are listed in the following table.

Time	0-4h	4-8h	8-12h	12-16h	16-20h	20-0h
Demand (GWh)	2	3	9	9	17	8

The producer has 6 coal power plants. 3 of them have a power of 1GW while the remaining 3 have a power of 2GW.
The cost of operating a 1GW plant is 100 €/GWh while the cost of operating a 2GW plant is 200 €/GWh.
There is a fixed cost for using a plant of 100 €per hour of use.
Finally, starting up a plant costs 400 €.

Type	Number	Power (GW)	Varying Cost €/GWh	Fixed Cost €/h	Startup Cost €
1	3	1	100	100	400
2	3	2	200	100	400

What is the optimal production planning to minimize the costs while satisfying the demand ?

Electricity production

An electricity producer wants to plan the level of production of his main power plants in order to fulfill the demand and minimize his costs. The demand for 6 time periods in the day are listed in the following table.

Time	0-4h	4-8h	8-12h	12-16h	16-20h	20-0h
Demand (GWh)	2	3	9	9	17	8

The producer has 6 coal power plants. 3 of them have a power of 1GW while the remaining 3 have a power of 2GW.
The cost of operating a 1GW plant is 100 €/GWh while the cost of operating a 2GW plant is 200 €/GWh.
There is a fixed cost for using a plant of 100 €per hour of use.
Finally, starting up a plant costs 400 €.

Type	Number	Power (GW)	Varying Cost €/GWh	Fixed Cost €/h	Startup Cost €
1	3	1	100	100	400
2	3	2	200	100	400

What is the optimal production planning to minimize the costs while satisfying the demand ?

Electricity production

An electricity producer wants to plan the level of production of his main power plants in order to fulfill the demand and minimize his costs. The demand for 6 time periods in the day are listed in the following table.

Time	0-4h	4-8h	8-12h	12-16h	16-20h	20-0h
Demand (GWh)	2	3	9	9	17	8

The producer has 6 coal power plants. 3 of them have a power of 1GW while the remaining 3 have a power of 2GW.
The cost of operating a 1GW plant is 100 €/GWh while the cost of operating a 2GW plant is 200 €/GWh.
There is a fixed cost for using a plant of 100 €per hour of use.
Finally, starting up a plant costs 400 €.

Type	Number	Power (GW)	Varying Cost €/GWh	Fixed Cost €/h	Startup Cost €
1	3	1	100	100	400
2	3	2	200	100	400

What is the optimal production planning to minimize the costs while satisfying the demand ?

Electricity production

An electricity producer wants to plan the level of production of his main power plants in order to fulfill the demand and minimize his costs. The demand for 6 time periods in the day are listed in the following table.

Time	0-4h	4-8h	8-12h	12-16h	16-20h	20-0h
Demand (GWh)	2	3	9	9	17	8

The producer has 6 coal power plants. 3 of them have a power of 1GW while the remaining 3 have a power of 2GW.
The cost of operating a 1GW plant is 100 €/GWh while the cost of operating a 2GW plant is 200 €/GWh.
There is a fixed cost for using a plant of 100 €per hour of use.
Finally, starting up a plant costs 400 €.

Type	Number	Power (GW)	Varying Cost €/GWh	Fixed Cost €/h	Startup Cost €
1	3	1	100	100	400
2	3	2	200	100	400

What is the optimal production planning to minimize the costs while satisfying the demand ?

Sequencing jobs on a bottleneck machine

In workshops, it may happen that a single machine determines the throughput of the production → critical machine.

A set of tasks is to be processed. The execution is non-preemptive. For every task i, a release date and duration are given.

Different possible objectives : total processing time (makespan), average processing time, total tardiness (if due dates are given)

Scheduling of telecommunications via satellite

A digital telecommunications system via satellite consists of a satellite and a set of stations on earth.

We consider for example 4 stations in the US that communicate with 4 stations in Europe through the satellite. The total traffic is given through a matrix $TRAF_{tr}$.

The satellite has a switch that allows any permutation between the 4 transmitters and the 4 receivers.

The cost of a switch is the duration of its longest packet.

The objective is to find a schedule with minimal total duration.

Scheduling nurses

A working day in a hospital is subdivided in 12 periods of two hours.
The personnel requirements change from period to period.

A nurse works 8 hours a day and is entitled to a break after having worked for 4 hours.

Determine the minimum number of nurses required to cover all requirements ?

If only 80 nurses are avalable, and assuming that it is not enough, we may allow for a certain number of nurses to work for a fifth period right after the last one. Determine a schedule the minimum number of nurses working overtime.