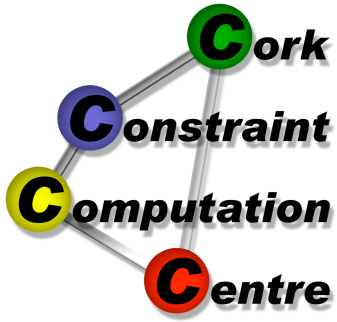


ROADEF Challenge 2010

Solving a Large-Scale Energy Management Problem:

A Constraint Programming and Greedy approach

Hadrien Cambazard, Emmanuel Hebrard,

Barry O'Sullivan

Cork Constraint Computation Centre (4C), Ireland

Outline of the talk

- Problem description

  Scheduling / Refueling / Production
- Our approach
  - General view
  - Computing feasible schedules
  - Computing good schedules
  - Planning the production campaigns
- Results
- Weaknesses

Problem Description

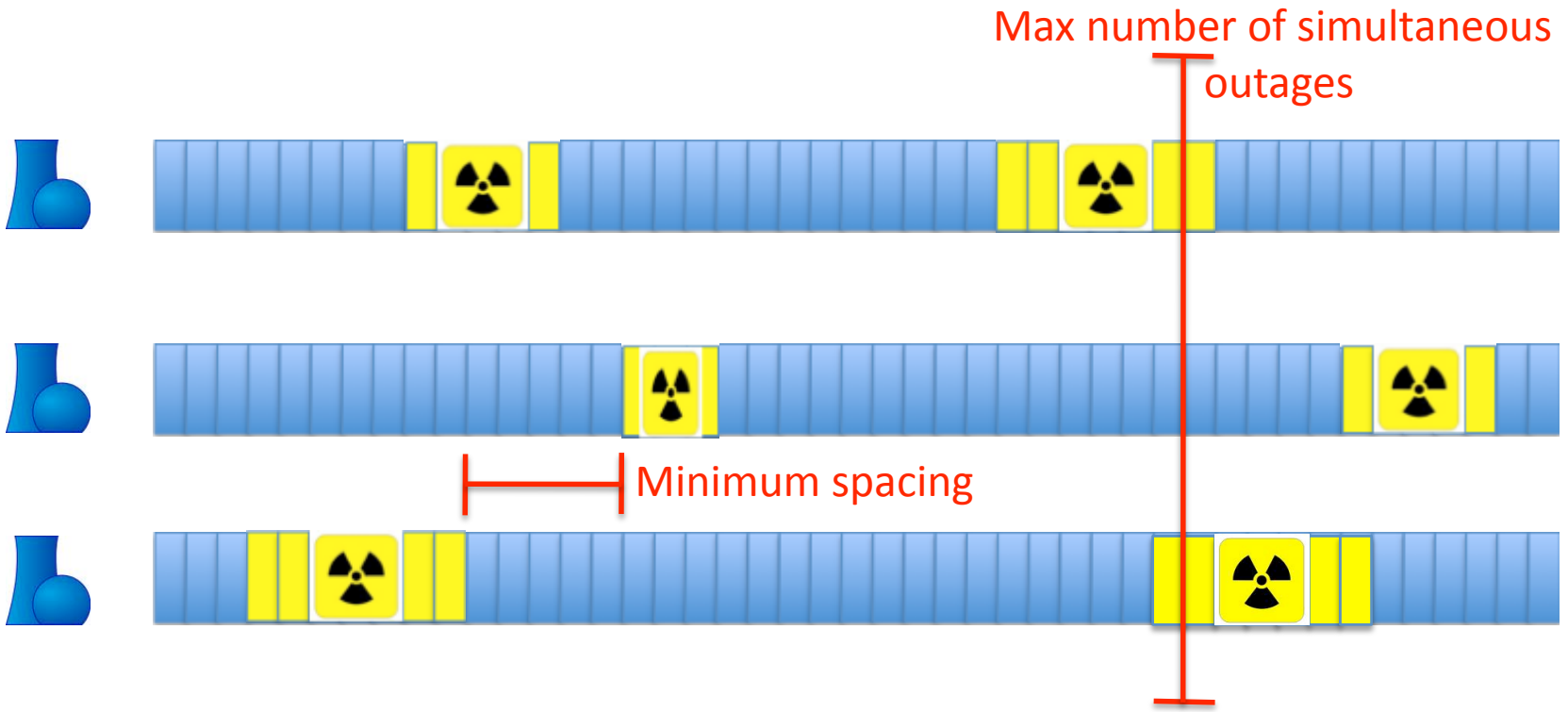
Management of a park of nuclear plants :
scheduling of shutdowns / refueling / production

- Problem stochastic with a scenario based model
- A solution to one scenario of the management problem :

  Nuclear plants :
  - Times of refueling (plants are put in outages)
  - Quantities of fuel
  - Level of production/stock for each time step

  Other plants (hydraulic) :
  - Level of production at each time step

Problem description - Scheduling

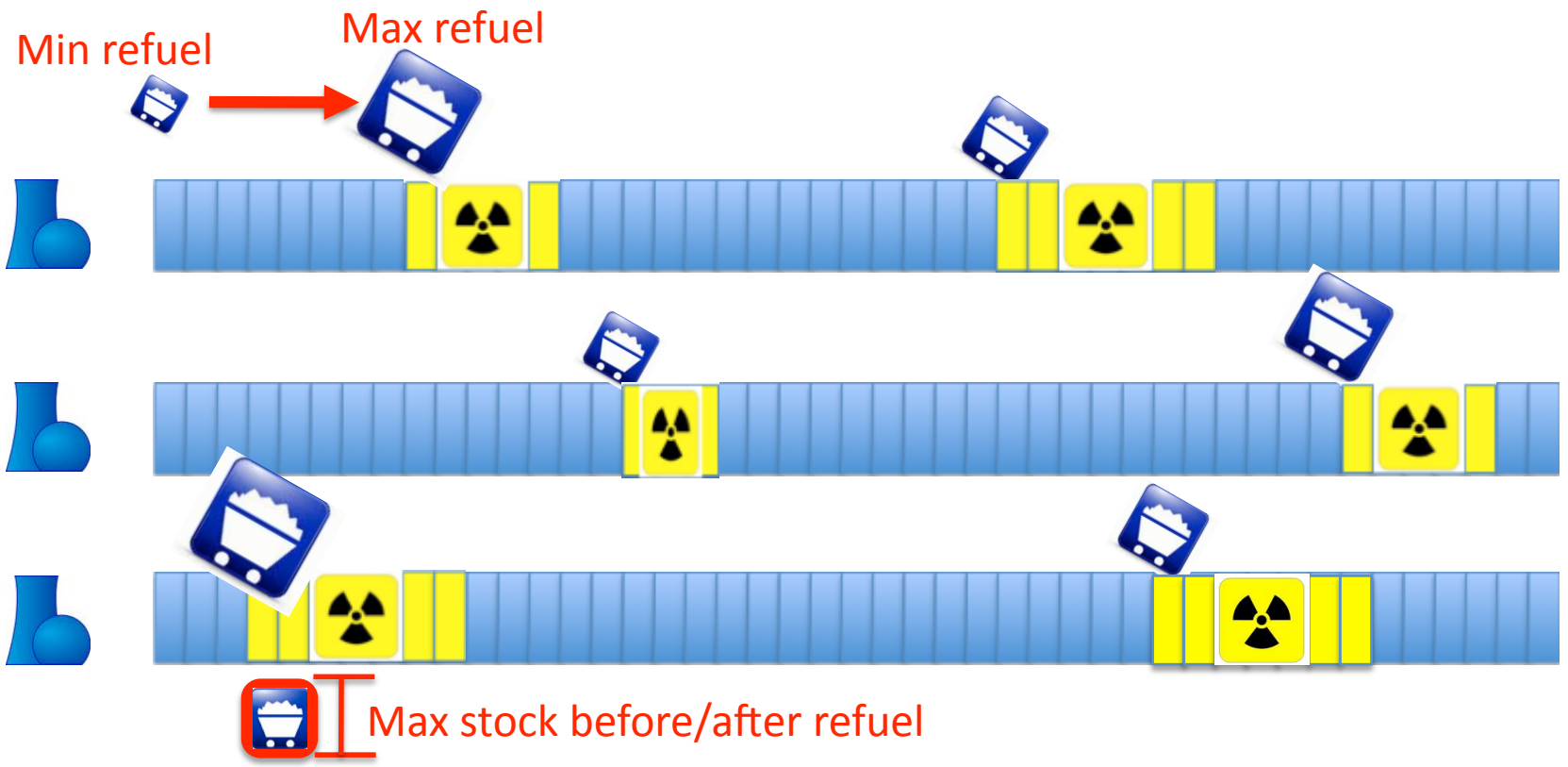


Scheduling :

Plants are put in outages (shutdowns) during refueling subject to :

- Disjunctive constraints (sometimes conditioned by the time of outages)
- Cumulative constraints

Problem description - Refueling



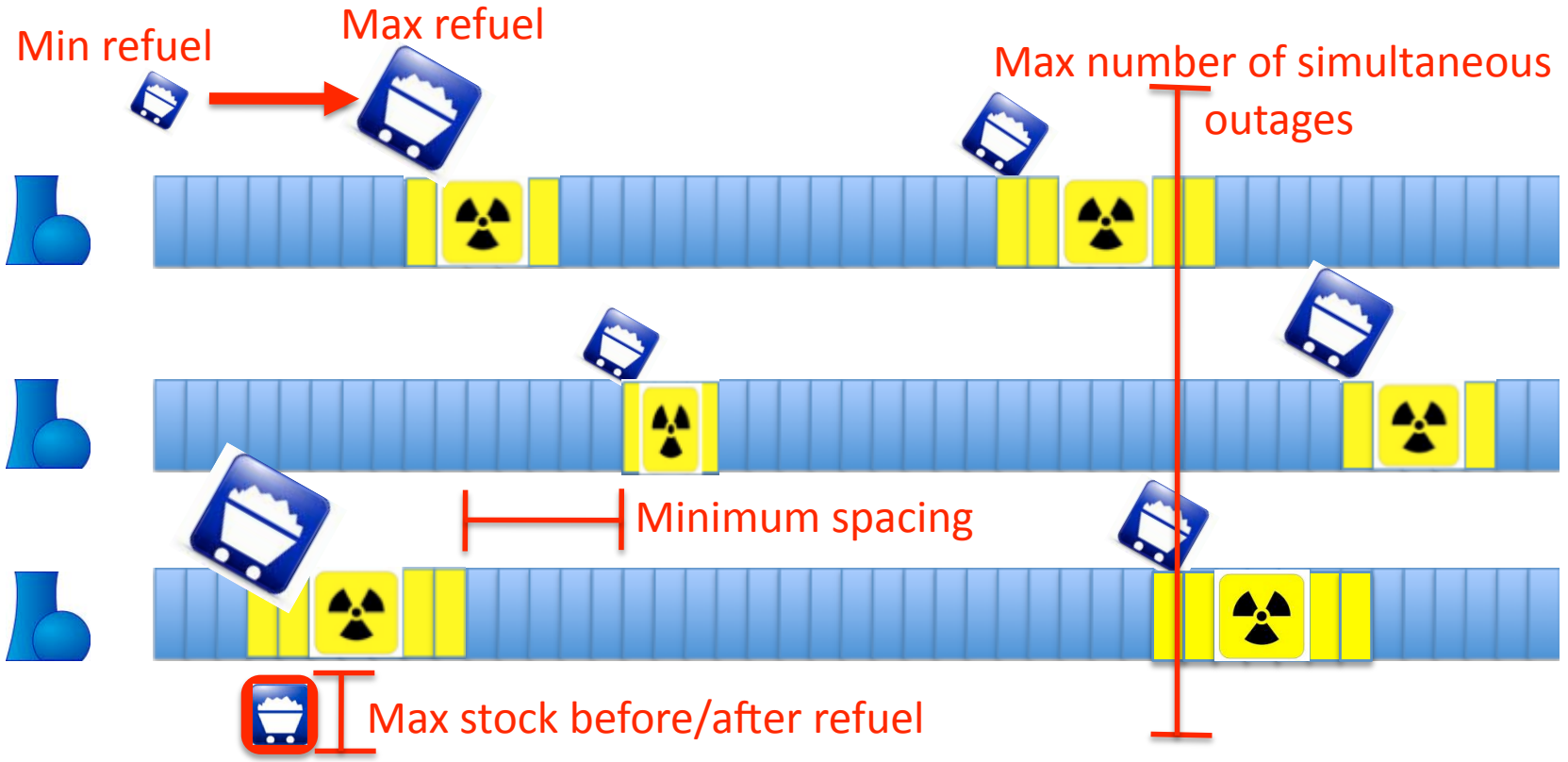
Refueling :

Refueling is subject to

Lower/Upper bounds of fuel

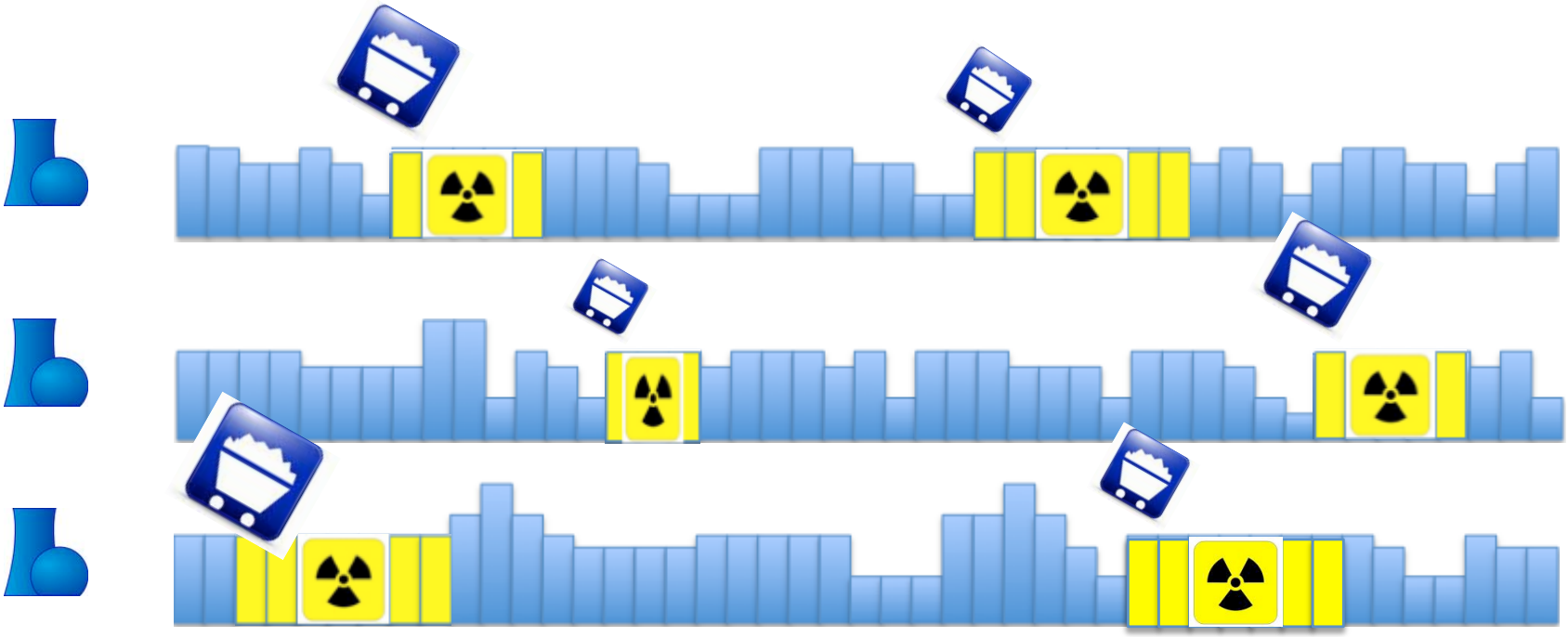
Upper bounds stated on the stock levels (before/after refueling)

Problem description – Scheduling/Refueling



Summary Scheduling/Refueling

Problem description – Production/stock



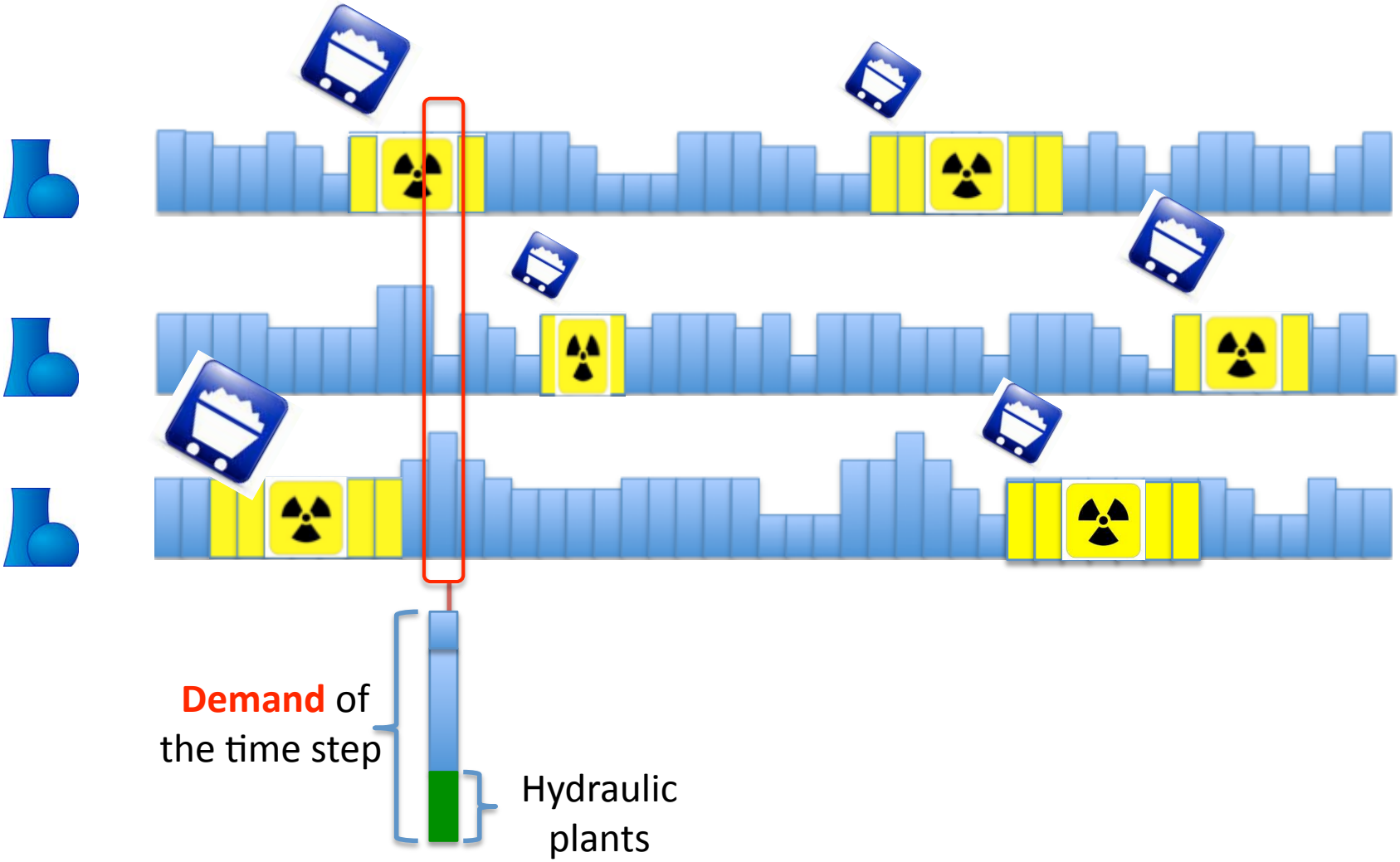
Scenario i

Production - Stock :

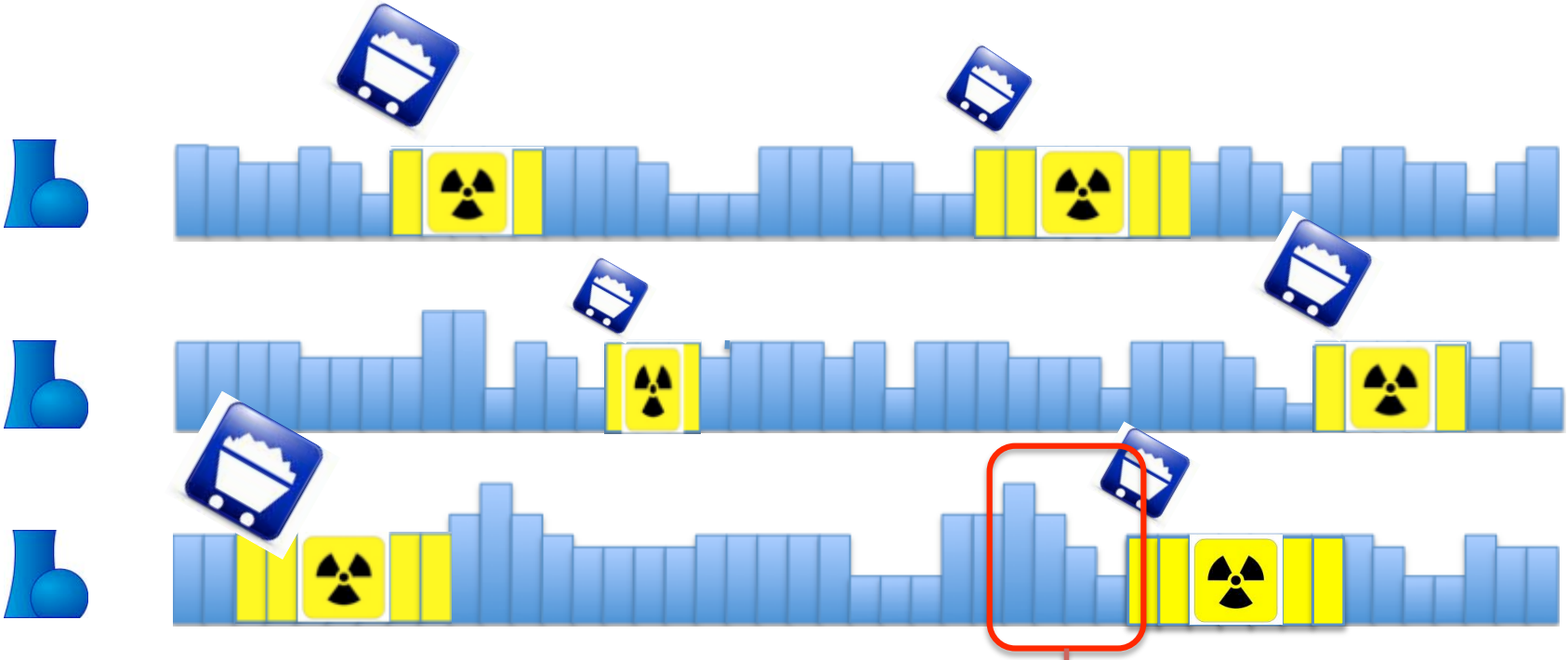
Level of production must be decided at each time step of each scenario

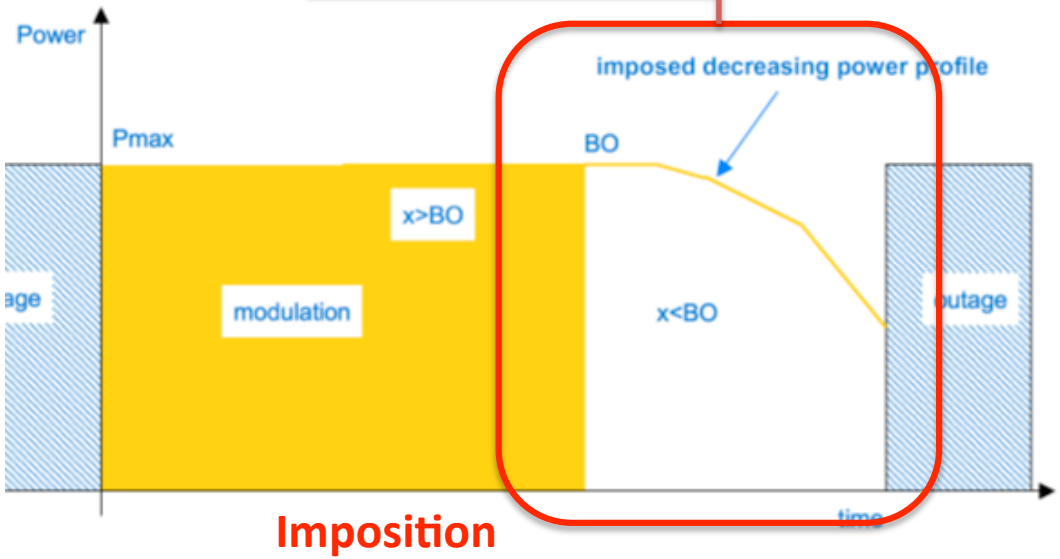
Once outages and refueling are known, each scenario is independent

Problem description – Production/stock



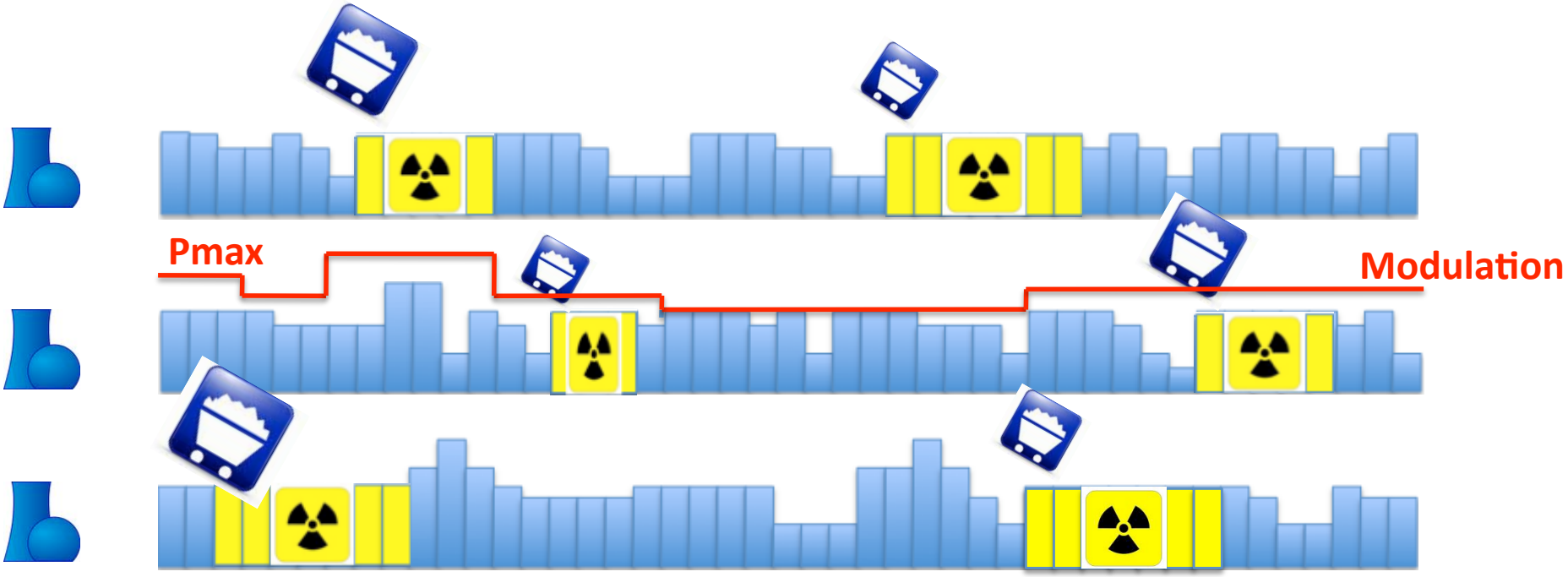
Problem description – Production/stock





Imposition

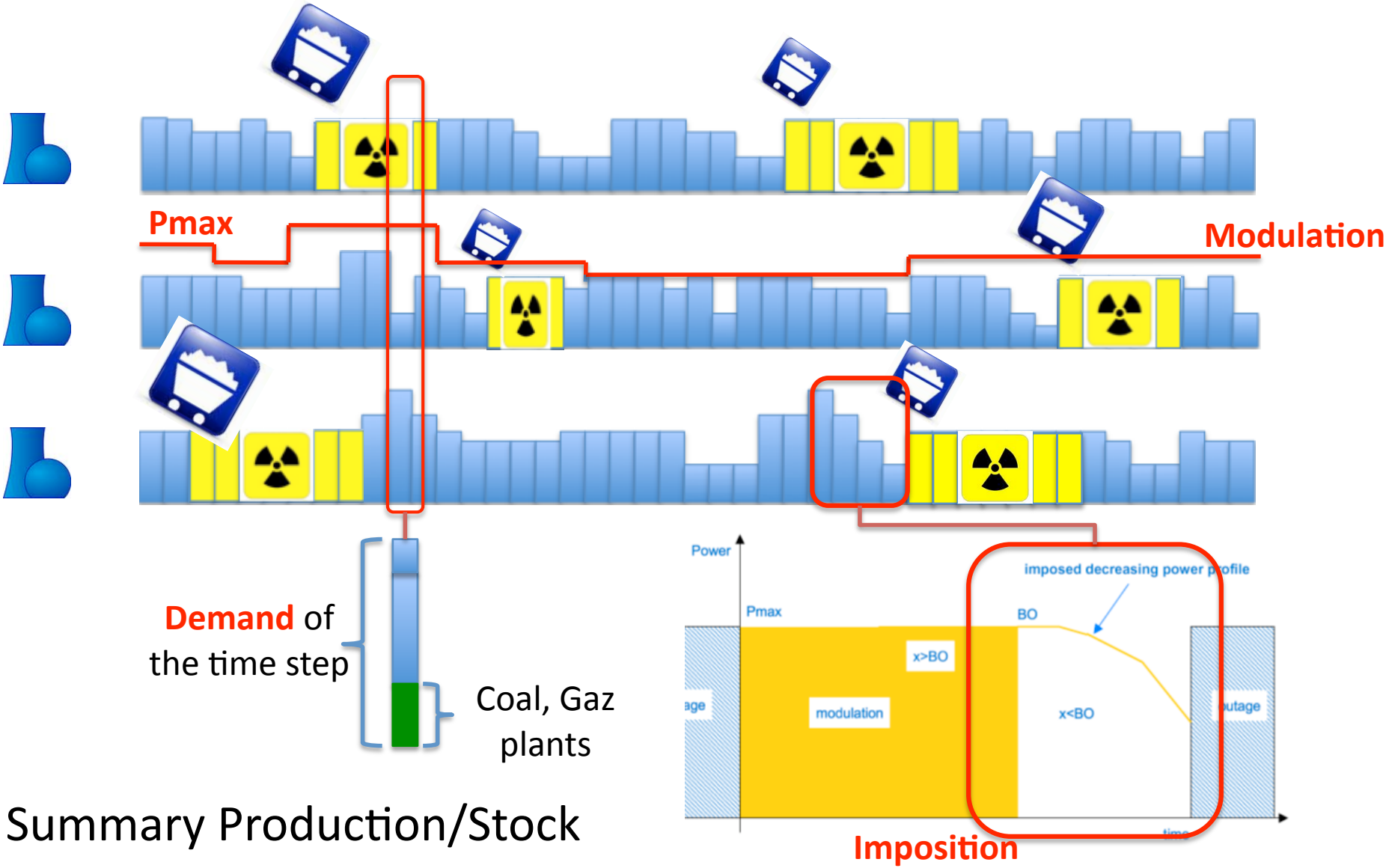
Problem description – Production/stock



Modulation constraint :

Enforce a minimum production (the gap to the maximum production profile is bounded)

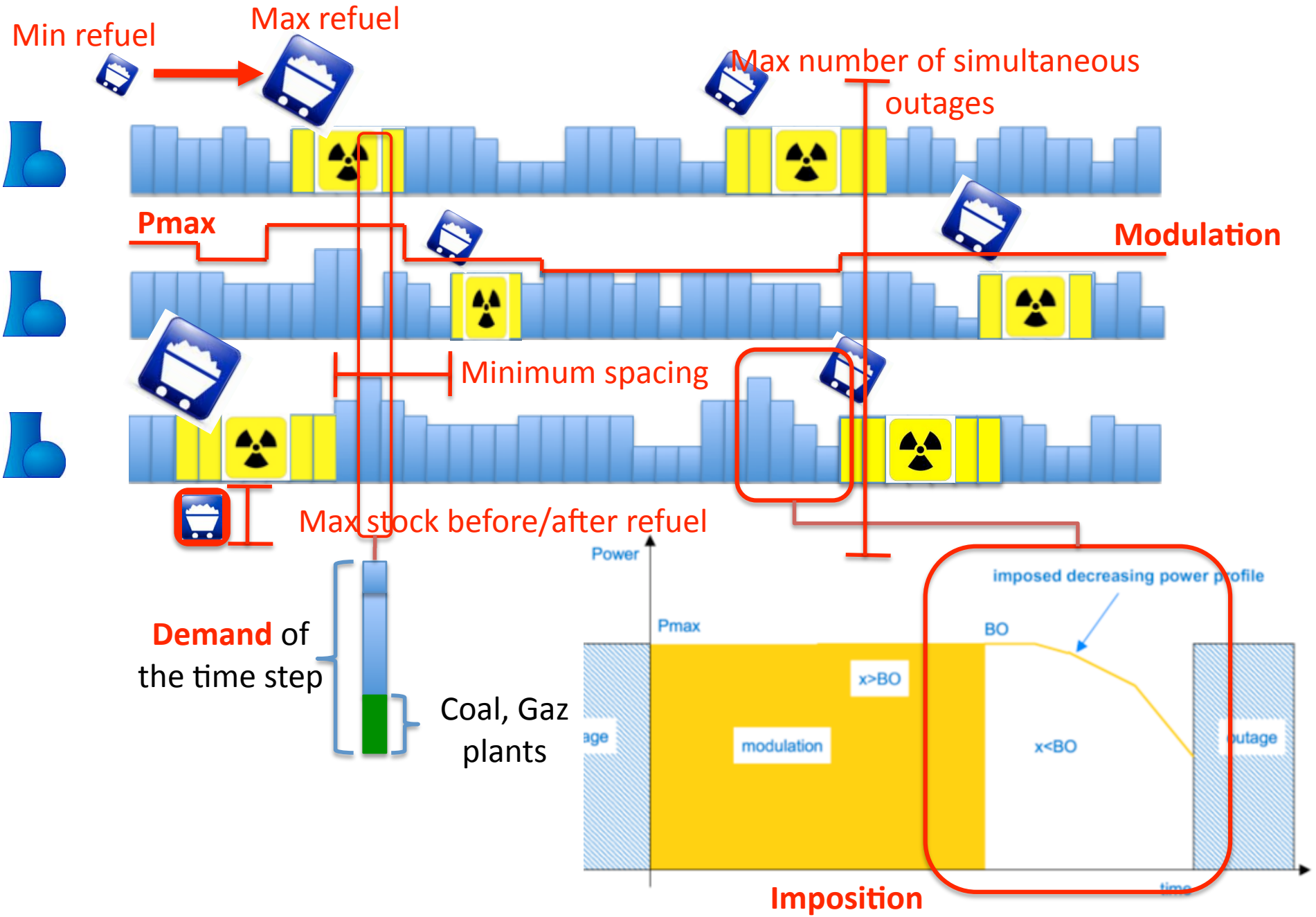
Problem description - Production / stock



Summary Production/Stock

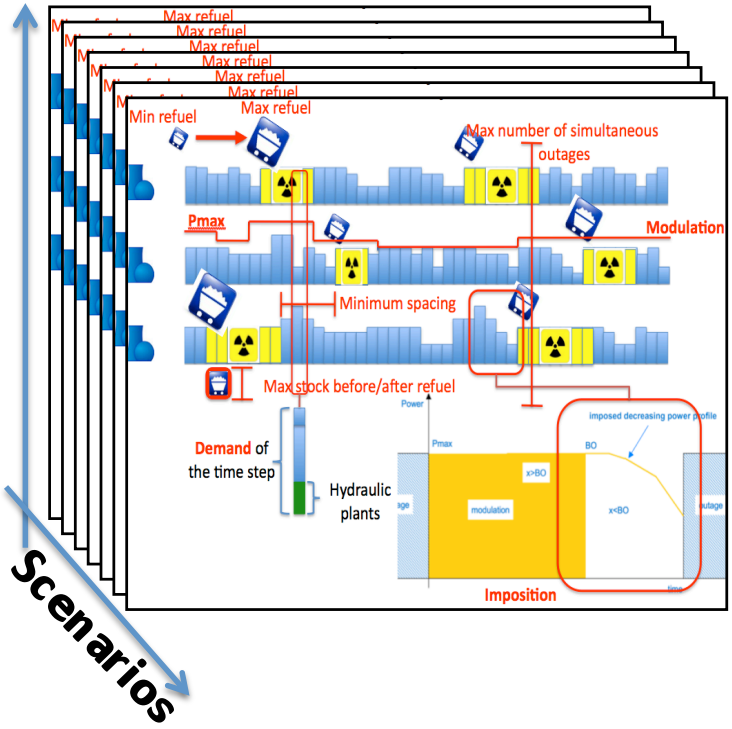
Plants are independent except for the demand constraint

Scheduling/Refueling/Production



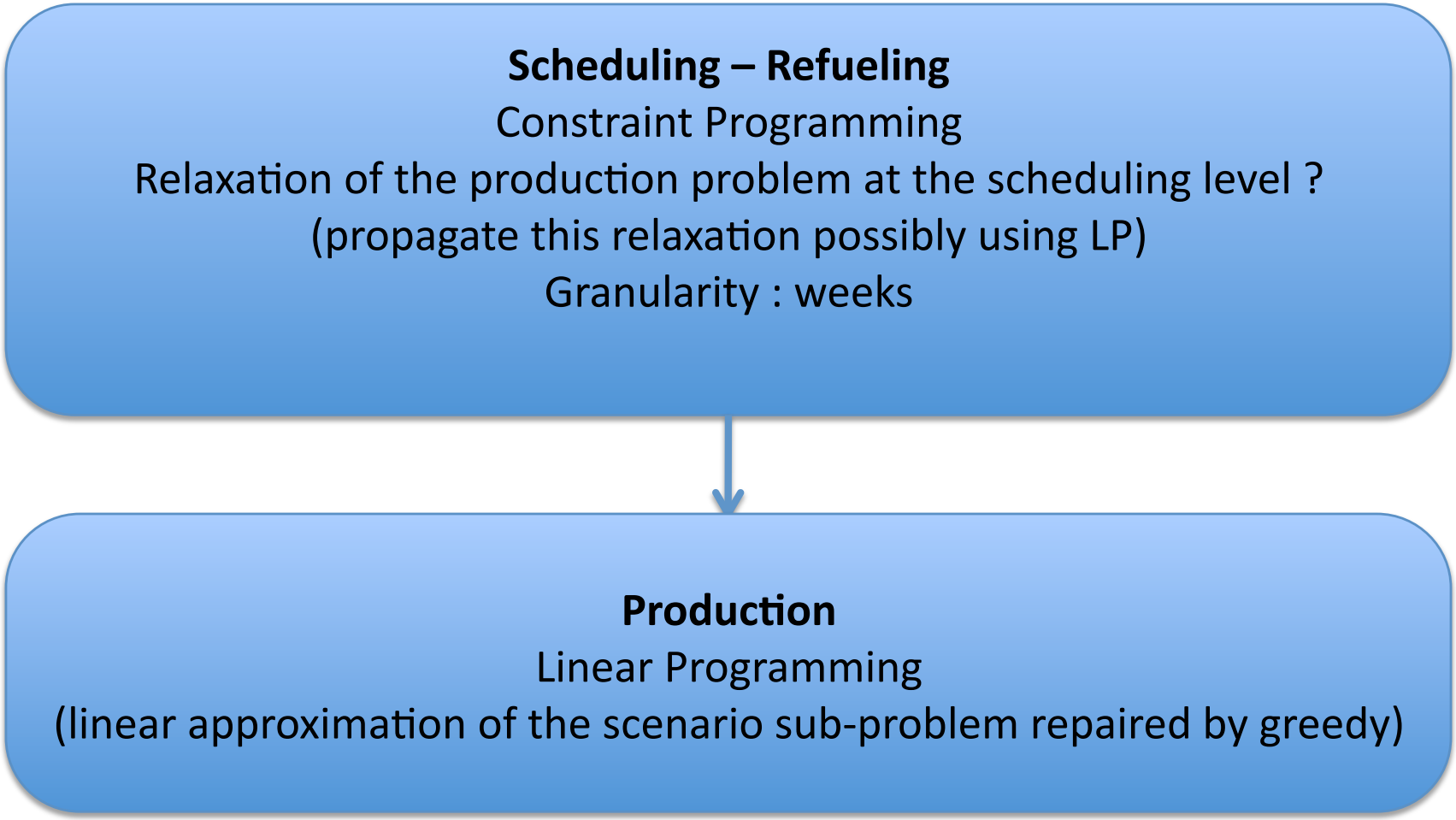
Problem size

- Typical problem size :
  - 56 nuclear plants / 20 hydraulic plants
  - 5000 – 6000 timesteps
  - 121 scenarios (up to 500)

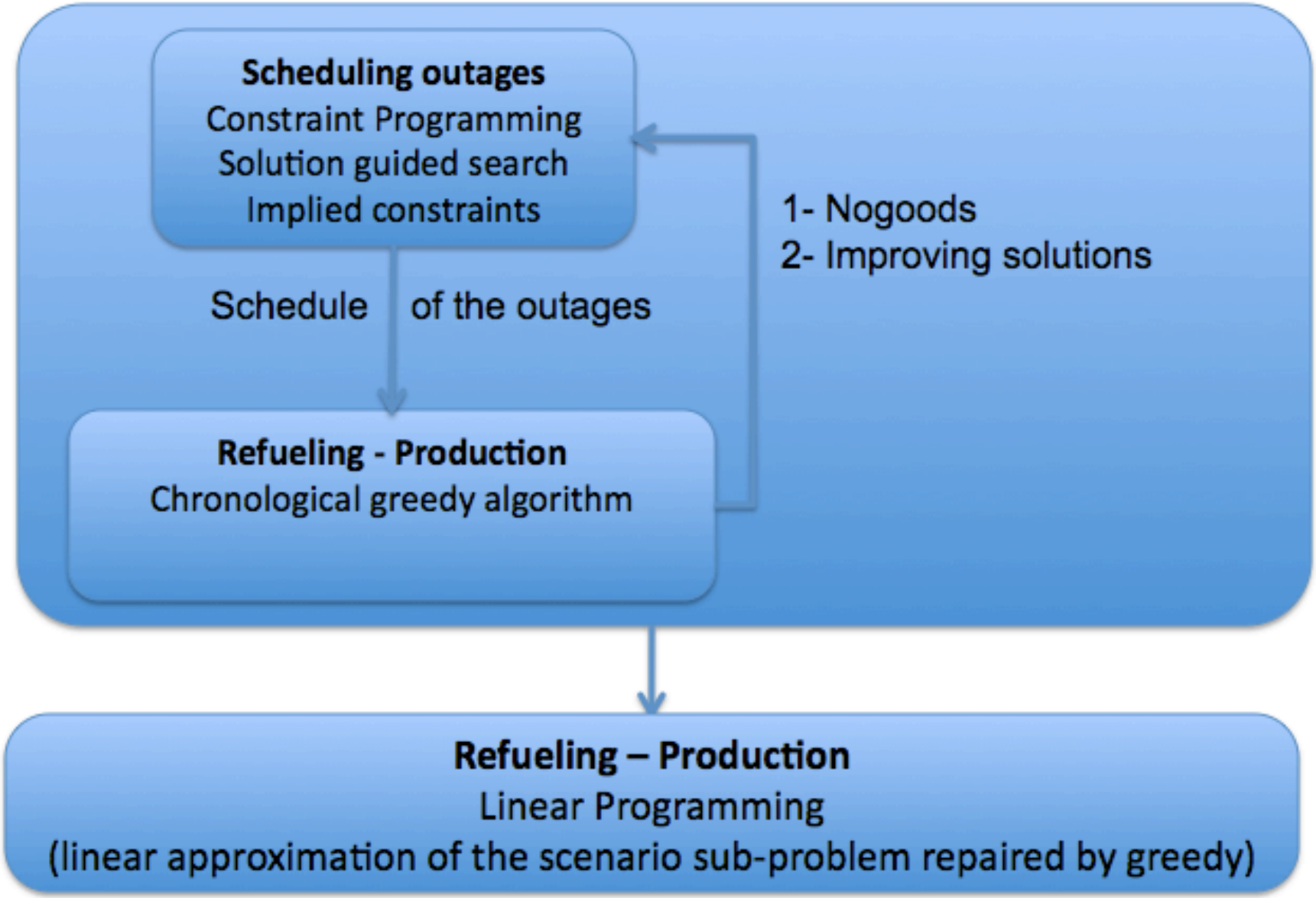


- In practice :
  - Size of a solution file : around 1 Go
  - Number of decision variables (dataB10) : more than 50,000,000
  - Refueling-Production : Large continuous domains
  - Scheduling : Discrete domains

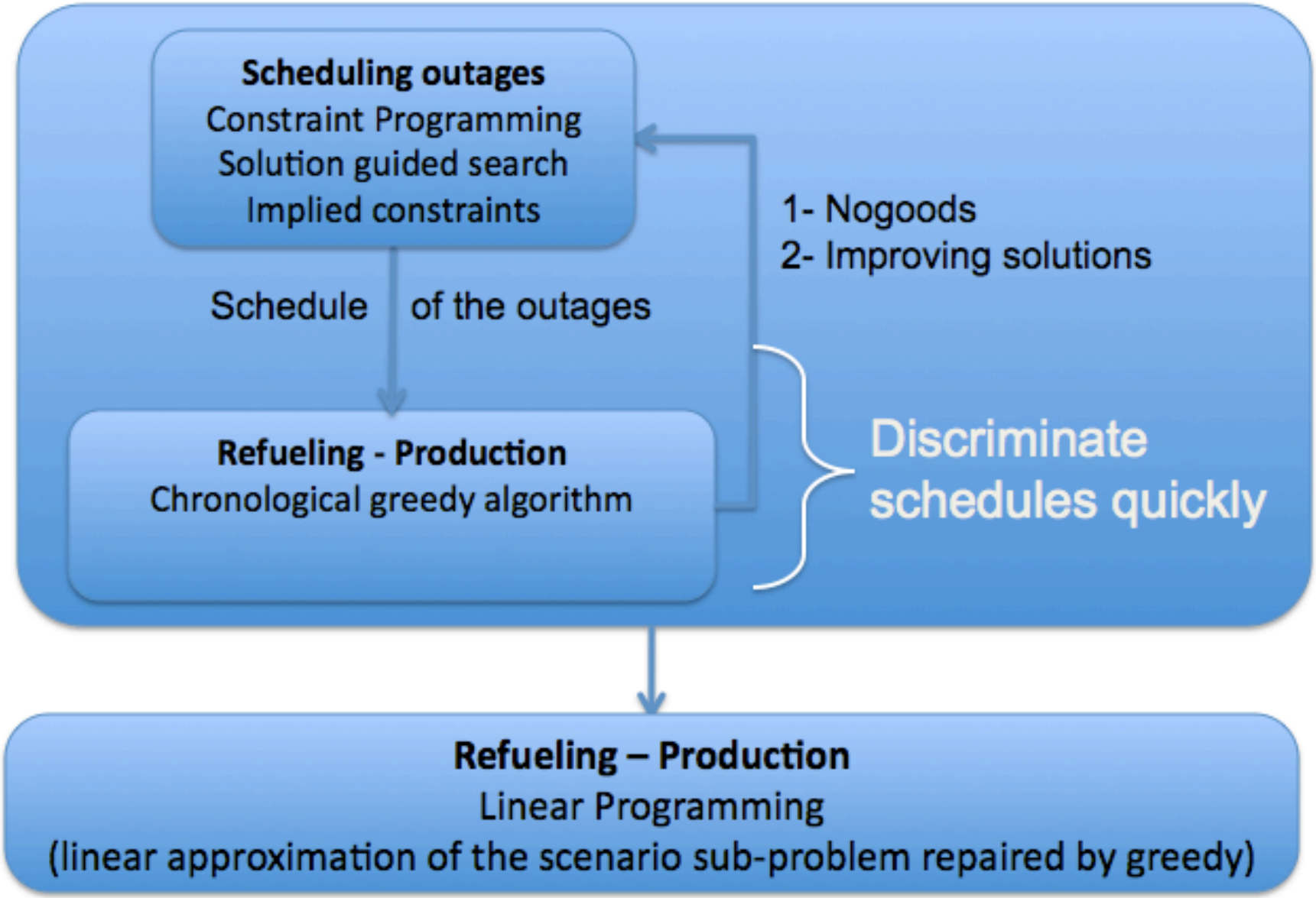
Our initial view



General view



General view

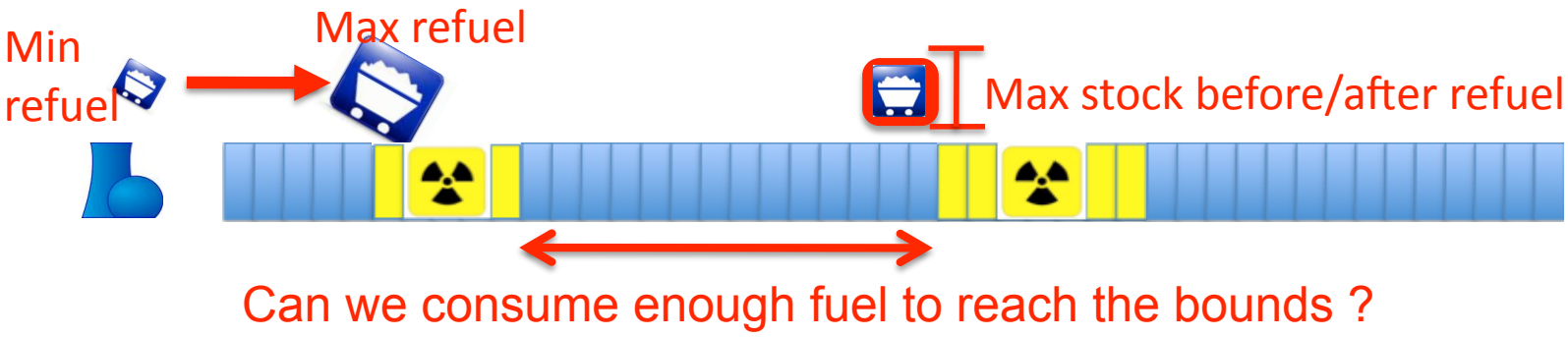


Computing feasible schedules

- Given a schedule of the outages
- Is it possible to refuel this schedule ? = Can the upper bounds on fuel at each outage be satisfied ?

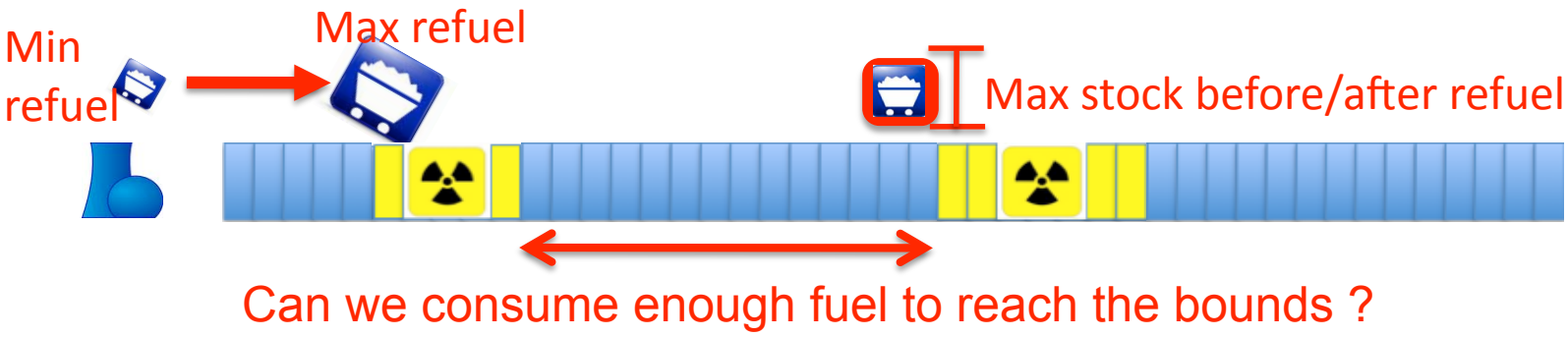
Computing feasible schedules

- Given a schedule of the outages
- Is it possible to refuel this schedule ? = Can the upper bounds on fuel at each outage be satisfied ?



Computing feasible schedules

- Given a schedule of the outages
- Is it possible to refuel this schedule ? = Can the upper bounds on fuel at each outage be satisfied ?



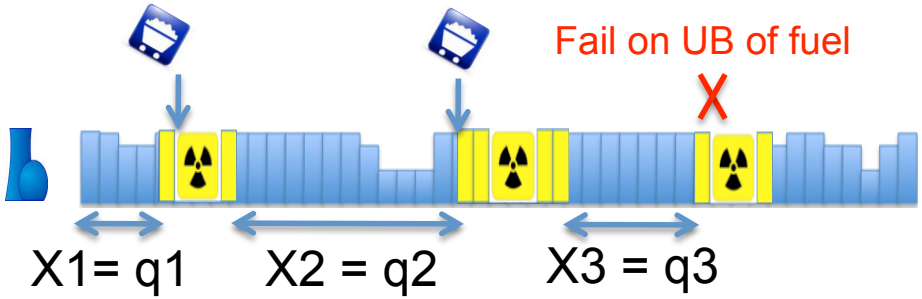
- Simulate the most optimistic campaign : refuel to the minimum (Rmin) and produce to the maximum (Pmax).
- Complicating factors :

  All plants can't produce to their max (because of the demand)

Computing feasible schedules

- (1) Solve the disjunctive/cumulative problem using CP

- (2) Check the feasibility of the refueling :
  - (2a) Check each plant independently (ignoring the demand constraint) with a min refuel – max production strategy
  - Return an explanation in case of infeasibility :



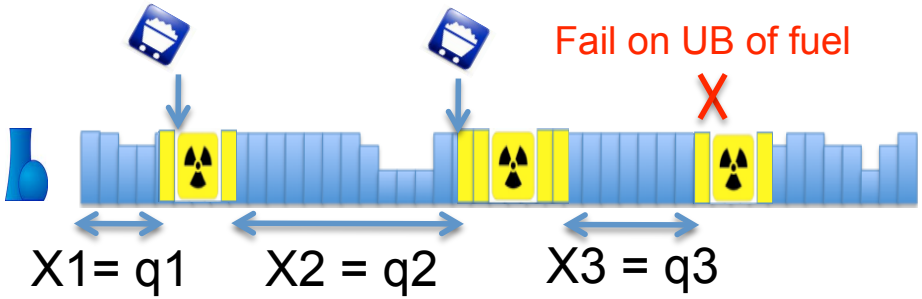
$$X1 + X2 + X3 > (q1 + q2 + q3) + Q$$

Computing feasible schedules

- (1) Solve the disjunctive/cumulative problem using CP

  ⇒ Add implied constraints for each outage Xi > q

- (2) Check the feasibility of the refueling :
  - (2a) Check each plant independently (ignoring the demand constraint) with a min refuel – max production strategy
  - Return an explanation in case of infeasibility :



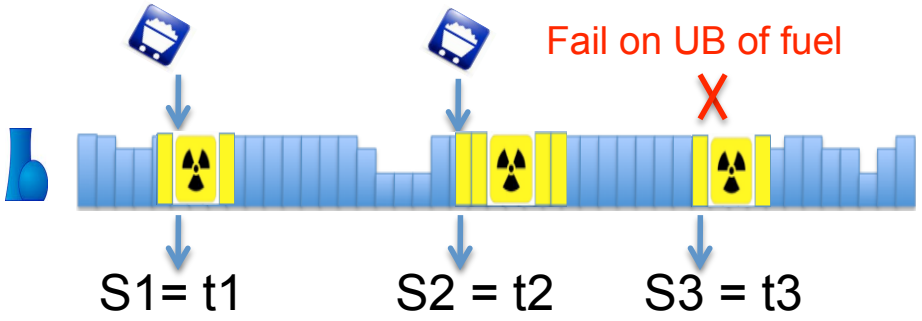
$$X1 + X2 + X3 > (q1 + q2 + q3) + Q$$

Computing feasible schedules

- (1) Solve the disjunctive/cumulative problem using CP

  → Add implied constraints for each outage Xi > q

- (2) Check the feasibility of the refueling :
  - (2a) Check each plant independently (ignoring the demand constraint) with a min refuel – max production strategy
  - Return an explanation in case of infeasibility :



→ (S1 != t1) V (S2 != t2) V (S3 != t3)

Computing feasible schedules

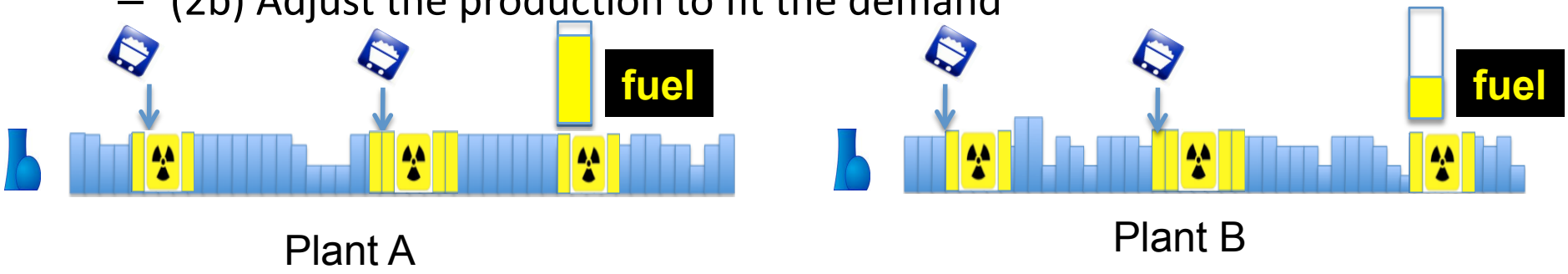
- (1) Solve the disjunctive/cumulative problem using CP

- (2) Check the feasibility of the refueling :
  - (2a) Check each plant independently (ignoring the demand constraint) with a min refuel – max production strategy
  - (2b) Adjust the production to fit the demand (min demand across all scenarios)

Computing feasible schedules

- (1) Solve the disjunctive/cumulative problem using CP

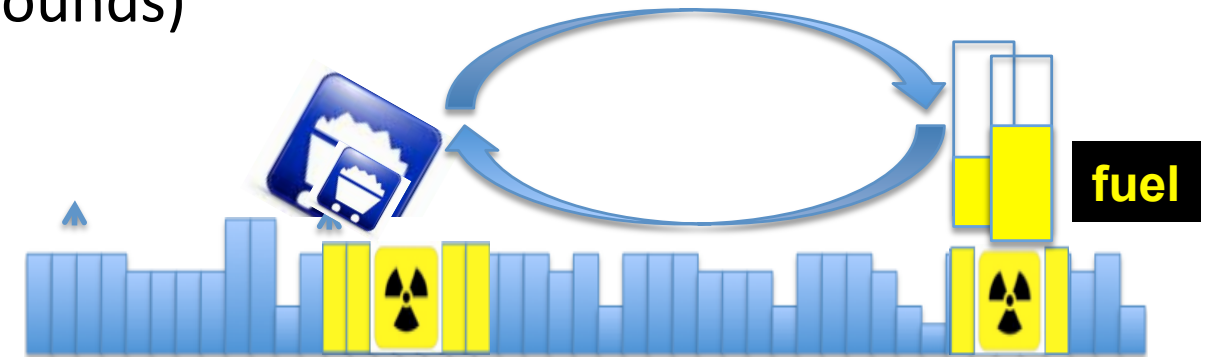
- (2) Check the feasibility of the refueling :
  - (2a) Check each plant independently (ignoring the demand constraint) with a min refuel – max production strategy :
  - (2b) Adjust the production to fit the demand



Plant A has to keep producing to its maximum capacity
Plant B has more flexibility than plant A

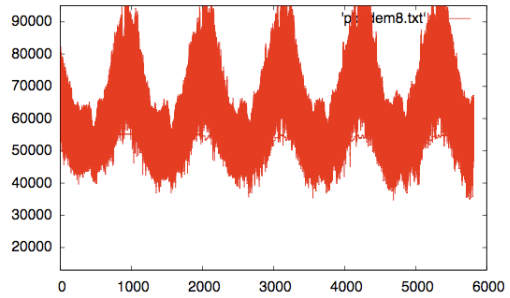
Computing feasible schedules

- (1) Solve the disjunctive/cumulative problem using CP
- (2) Check the feasibility of the refueling :
  - (2a) Check each plant independently (ignoring the demand constraint) with a min refuel – max production strategy
  - (2b) Adjust the production to fit the demand => **estimated production**
- (3) Increase the refueling to reach the estimated production (checking the fuel bounds)



Computing good schedules

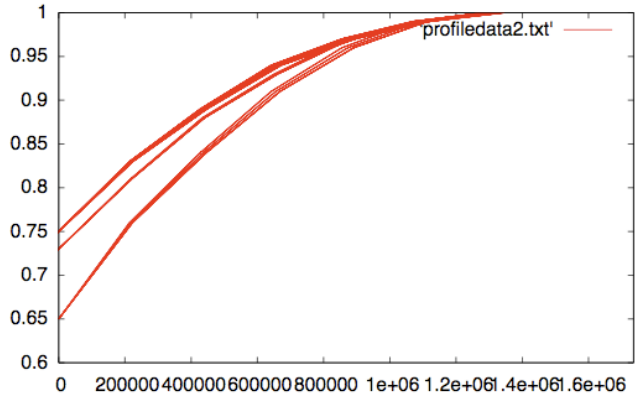
- Ideally : A relaxation of the refueling/production sub problem at the scheduling level.

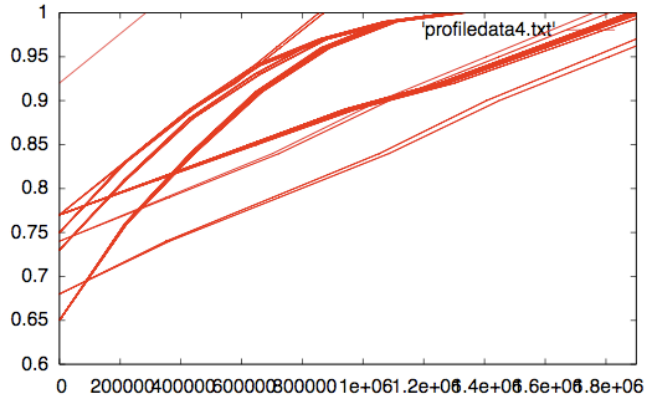


- In Practice, guide the search of the CP master using the best known schedule :
  - We branch (domain splitting) in the direction of the best known schedule with a probability **(1-P)**.
  - **P** is cooled to progressively force the CP solver to search around the best schedule.

Computing good production campaigns

- The problem is continuous and linear when relaxing :
  - One profile of the imposition constraint (The lower)
  - The non convex profiles





- Apply the same greedy procedure guided by the values of the LP (to eventually repair it)

Results on A set

- 30 minutes timelimit, 1 run (very small variance)
- Improvement given by the PL is significant

instances	Best known qualification cost	Cost greedy	Final cost (after PL)
A1	16954	16958	**16949**
A2	14605	14605	**14597**
A3	15443	15450	**15432**
A4	11159	11171	**11157**
A5	12582	12600	**12513**

Results on B set

- PL has to be relaxed further to scale and bring little improvement compared to set B
- 10 runs with different seeds

instances	Avg Cost	Min Cost	Avg Schedules	Avg Improving schedules	Avg Nb Nogoods
B6	87248	86991	1673	84	1215
B7	83286	82936	2345	107	3293
B8	231431	119240	4926	12	15745
B9	129613	109902	5920	14	19698
B10	82804	82330	726	29	240

The weaknesses of this approach

- Scheduling stage is blind regarding the refueling constraints (**Encode a fuel conservation within a global constraint including the imposition constraint)**
- Not enough **incremental** in the evaluation of a serie of schedules.
- Scenarios are the bottleneck (need to work on an **approximation of all scenarios**)
- LP do not scale for the production subproblem on large instances…

Conclusion

- CP with Solution guided search and addition of nogoods
- Greedy refueling/production to rank schedules
- LP for optimizing the production

- Implementation made in NumberJack:

http://4c110.ucc.ie/numberjack/