Du générique au spécifique : Problèmes d'ordonnancement avec production et consommation des ressources

Jacques CARLIER **Aziz MOUKRIM** **Abderrahim SAHLI**

Sorbonne universités, Université de Technologie de Compiègne, France
UMR CNRS 7253 - Laboratoire Heudiasyc

17ème Conférence ROADEF de la société Française de Recherche Opérationnelle et d'Aide à la décision

Compiègne, 10-12 Février 2016

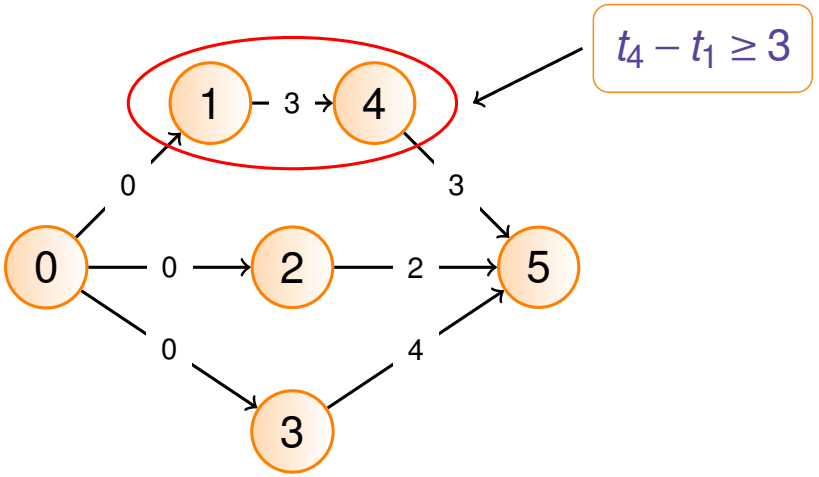
Plan

- Introduction
- ESPCPR
- Literature
- MILP for ESPCRP
- Strict Order Algorithm
- Classical problems
- Lower bounds
- Results
- Conclusion

Introduction

Scheduling Problems

- **Activities:** processing time, start time, completion time, release date, due date.
- **Precedence constraints:** represented by a precedence graph.



- **Resources:** renewable(machine), non-renewable (money, fuel).
- **Objective functions:** Makespan, total weighted completion time...

Publication:
Jeremy Rifkin (2011). The Third Industrial Revolution: How Lateral Power is Transforming Energy, the Economy, and the World. New York: Palgrave Macmillan, 270 p.

Introduction

Our problem

- **Resource Constrained Project Scheduling Problem (RCPSP):**
  - **Project:** Activities, Renewable Resources
  - **Constraints:** Potential Constraints, Resource Constraints
  - **Aim:** Minimize the project duration

- **Event Scheduling Problem with Consumption and Production of Resources (ESPCPR):**
  - ESPCPR ⇔ RCPSP + production and consumption events

Problem:
How to plan the project in order to minimize the makespan?

Introduction

- **Scheduling Problems with Renewable Resources (RCPSP):**
  - **Johnson's Rule:** Two machine flow-shop
  - **Jackson's Rule:** Sequencing problem with lateness minimization
  - List Scheduling Algorithm (heuristic)
  - Arbitrage Algorithm (exact method)

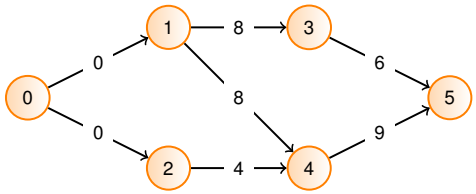
- **Scheduling Problems with Non-Renewable Resources:**
  - **Financing problem:** Produce resources as soon as possible, consume resources as late as possible

- **ESPCPR:** Generalization of the previous rules and algorithms

Introduction

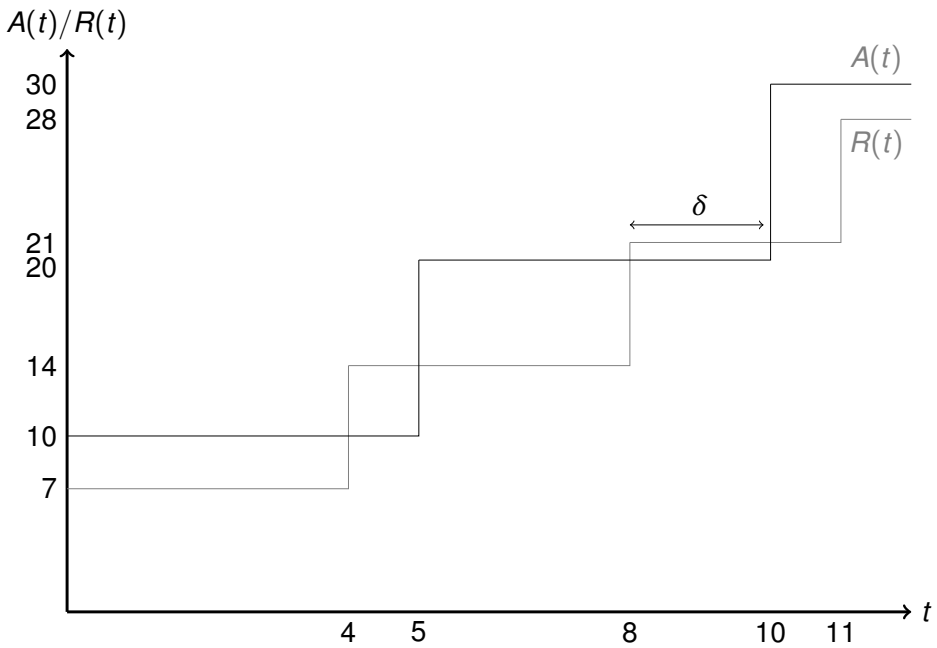
The Financing Problem

- The Financing Problem (Carlier and Rinnooy Kan 1982; Slowinski 1984)
  - **Production events:** The arrival dates are given
  - **Consumption events:** There are precedence constraints between them.



Example with four tasks such that $r_1 = r_2 = r_3 = r_4 = 7$ and $\tau_1 = 0$, $\tau_2 = 5$ and $\tau_3 = 10$, $b_1 = b_2 = b_3 = 10$

- **The shifting algorithm:** A polynomial algorithm to solve the financing problem



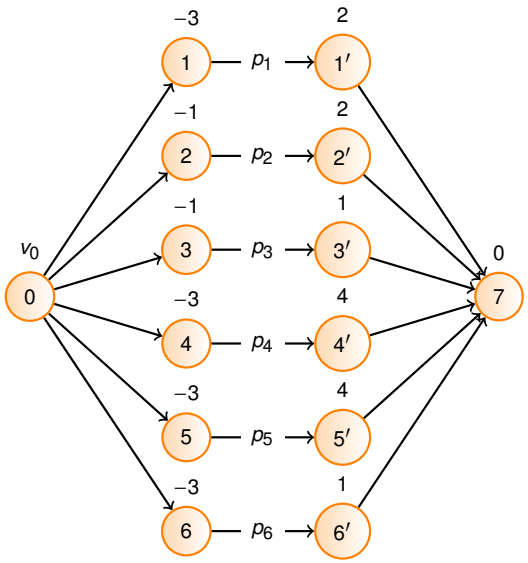
Applying the shifting algorithm

Introduction

The Relocation Problem

- Resource-constrained single-machine scheduling problem **(Lin et Cheng 1999).**
  - RP=(J, c, b, p, v_0)
  - J: Set of n jobs
  - c_i: The quantity of resource acquired by the job i
  - b_i: The quantity of resource returned by the job i
  - p_i: The processing time of the job i
  - v_0: The initial quantity of a single type of resource

- The relocation problem ⇔ A two-event parallel-chain problem.



An instance of the relocation problem represented by an instance of ESPCPR

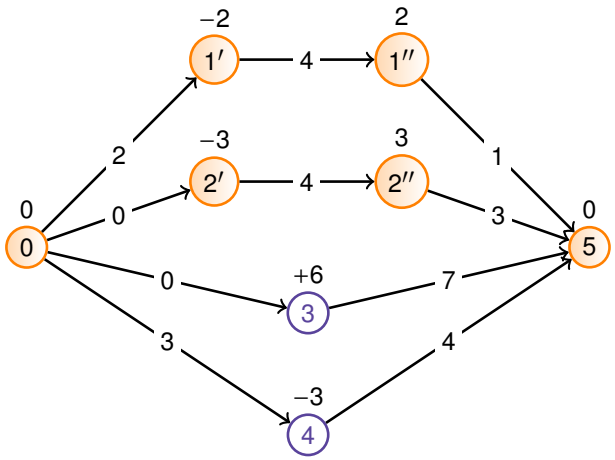
Introduction

The Generalized Cumulative Scheduling Problem

- **Notation:** GCuSP
- Generalization of CuSP
- Y: Set of non-preemptive activities
- X: Set of production and consumption events
- b_i: The number of resource units produced or consumed by event i
  - Positive: production
  - Negative: consumption
- **Aim:** Minimize the makespan

Operation	1	2	3	4
Release date	2	0	0	3
Processing time	4	4	0	0
Tail	1	3	7	4
Resource	2	3	+6	-3

An instance of GCuSP(Available resource = 0)



ESPCPR

Plan

- Introduction
- ESPCPR
- Literature
- MILP for ESPCRP
- Strict Order Algorithm
- Classical problems
- Lower bounds
- Results
- Conclusion

Event Scheduling Problem with Consumption and Production of Resources (ESPCPR)

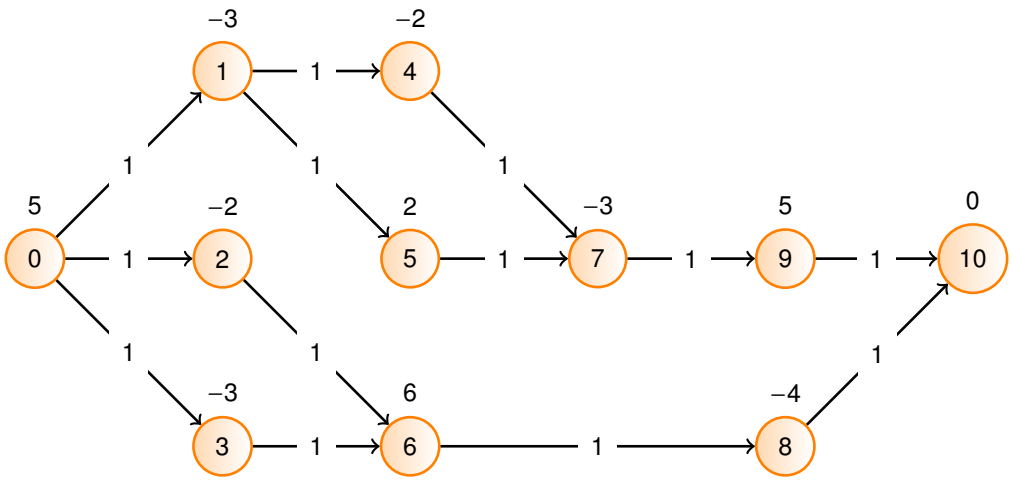
Our model

- Production and consumption events
- a_i^k: The quantity of resource k produced or consumed by event i
  - Positive: production
  - Negative: consumption
- S_i: The occurrence time of the event i
- v_{ij}: The time lag between event i and event j
  - Positive: maximum
  - Negative: minimum
- H: the scheduling horizon
- ES_i: the earliest occurrence time of event i
- LS_i: the latest occurrence time of event i

ESPCPR

Our model

- RCPSP + Production and Consumption events
- Events, Arcs=time lags, production and consumption of resources



An instance of the ESPCPR

Two problems

- Decision problem (NP-complete)
- Optimization problem (NP-hard)

Plan

- Introduction
- ESPCPR
- Literature
- MILP for ESPCRP
- Strict Order Algorithm
- Classical problems
- Lower bounds
- Results
- Conclusion

Existing models

- The financing problem (Carlier and Rinnooy Kan, 1982)
- The relocation problem (Kaplan and Amir, 1988; Lin and Cheng, 1999)
- Series-Parallel Precedence Constraints (Abdel-Wahab and Kameda, 1978)
- Project Scheduling with inventory constraints (Neumann and Schwindt, 2001)
- Resource Temporal Network (Laborie, 2003)

Publications:

- H.M. Abdel-wahab, T. Kameda, Scheduling to minimize maximum cumulative cost subject to series-parallel precedence constraints, Operations Research 26 (1) (1978) 141-158.
- J. Carlier and A.H.G. Rinnooy Kan, Financing and Scheduling, OPER RES LETT, (1982), Vol 1, 52-55.
- P. Laborie, Resource Temporal Networks: Definition and Complexity, IJCAI (2003) 948-953.
- E.H. Kaplan, A. Amir, A fast feasibility test for relocation problems, European Journal of Operational Research 35 (1988) 201-205.
- B.M.T. Lin, T.C.E. Cheng, Relocation problems to minimize the maximum tardiness and the number of tardy jobs, European Journal of Operational Research 116 (1999) 183-193.
- K. Neumann, C. Schwindt, Project scheduling with inventory constraints, Mathematical Methods of Operations Research 56 (2002) 513-533.

Existing models

- Generalized model including consumption and production of resources (Bouly 2004; Beldiceanu and Poder 2004)
- Generalized Resource-Constrained Project Scheduling Problem (Carlier, Moukrim and Xu 2009)
- Resource-Constrained Project Scheduling Problem with production and consumption of resources (Koné and al 2013)

Publications:

- N. Beldiceanu and E. Poder. A Continuous Multi-Resources Cumulative Constraint with Positive - Negative Resource Consumption - Production. In CPAIOR, pages 214-228, 2007.
- H. Bouly. Généralisation d'algorithmes d'ordonnancement à la production de ressources par les tâches, Rapport de DEA. 2004.
- J. Carlier, A. Moukrim and H. Xu. The Project Scheduling Problem with Production and Consumption of Resources: a list-scheduling based algorithm. Discrete Applied Mathematics, GO VI, 2009.
- O. Koné, C. Artigues, P. Lopez, and M. Mongeau. Comparison of mixed integer linear programming models for the resource-constrained project scheduling problem with consumption and production of resources, Flex Serv Manuf J (2013) 25:25-47

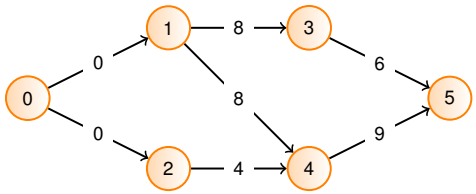
The Financing Problem

- Special case of the ESPCPR

  (Carlier and Rinnooy Kan 1982;

  Slowinski 1984)

  - **Production events:** The arrival dates are given
  - **Consumption events:** There are precedence constraints between them.



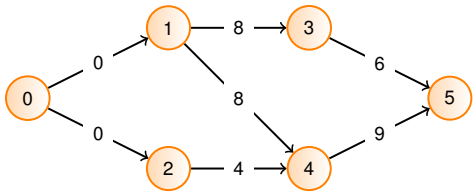
Example with four tasks such that $r_1 = r_2 = r_3 = r_4 = 7$ and $\tau_1 = 0$, $\tau_2 = 5$ and $\tau_3 = 10$, $b_1 = b_2 = b_3 = 10$

- **The shifting algorithm:** A polynomial algorithm to solve the financing problem

Applying the shifting algorithm

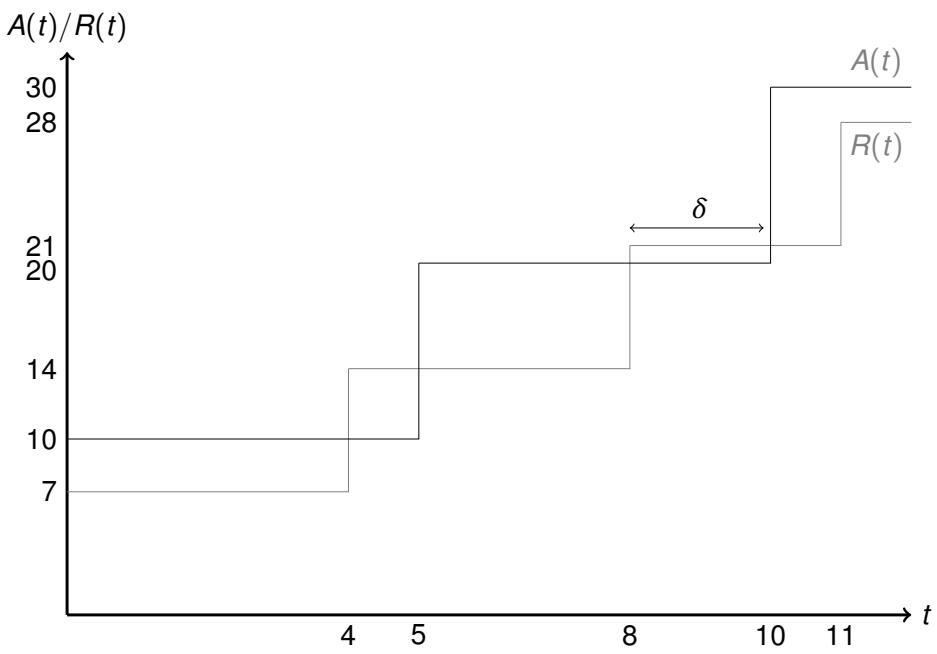
The Financing Problem

- Special case of the ESPCPR **(Carlier and Rinnooy Kan 1982; Slowinski 1984)**
  - **Production events:** The arrival dates are given
  - **Consumption events:** There are precedence constraints between them.



Example with four tasks such that $r_1 = r_2 = r_3 = r_4 = 7$ and $\tau_1 = 0$, $\tau_2 = 5$ and $\tau_3 = 10$, $b_1 = b_2 = b_3 = 10$

- **The shifting algorithm:** A polynomial algorithm to solve the financing problem



Applying the shifting algorithm

The Financing Problem

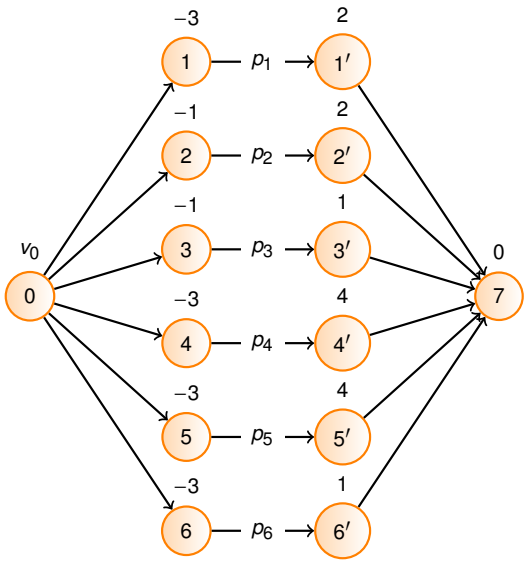
The Financing Problem extension

- Generalization of the shifting algorithm to other models of production and consumption of resources
- Complexity of problems with renewable and non-renewable resources = generally NP-Hard

The Relocation Problem

- Resource-constrained single-machine scheduling problem **(Lin et Cheng 1999).**
  - RP=(J, c, b, p, v_0)
  - J: Set of n jobs
  - c_i: The quantity of resource acquired by the job i
  - b_i: The quantity of resource returned by the job i
  - p_i: The processing time of the job i
  - v_0: The initial quantity of single type of resource
- The relocation problem and ESPCPR
  - The relocation problem RP=(J, c, b, p, v_0) can be represented by an instance of ESPCPR $I = (X, U, a, v)$
  - $X = \{1, 1', 2, 2', ..., n, n'\}$
  - $a_0 = v_0 \quad a_i = -c_i \quad a_{i'} = b_i$
  - $v_{ii'} = p_i$

- The relocation problem ⇔ A two-event parallel-chain problem.

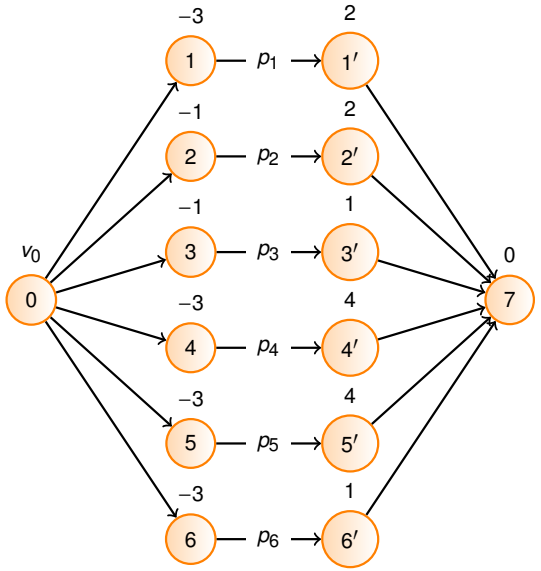


An instance of the relocation problem represented by an instance of ESPCPR

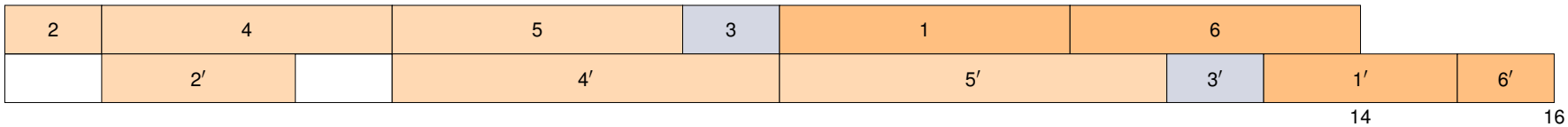
The Relocation Problem

The decision problem

- Equivalent to the two-machine flowshop problem
  - The resource corresponds to the idle time on the second machine in the flowshop
  - c_i: Processing time on the first machine
  - b_i: Processing time on the second machine
- Resolution by the Johnson's rule $O(n \log n)$
  - $N = \{i \in J / c_i < b_i\}$, $L = \{i \in J / c_i = b_i\}$ and $P = \{i \in J / c_i > b_i\}$
  - Schedule the jobs in N in non-decreasing order of c_i, the jobs in L in any order, and the jobs in P in non-increasing order of b_i
  - Concatenate the three sequences(N.L.P)



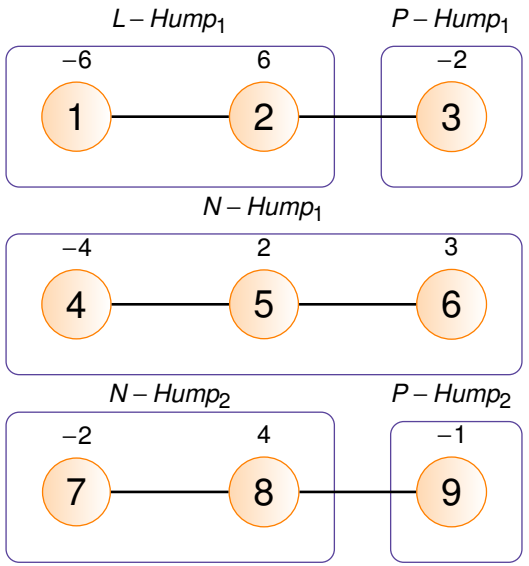
An instance of the relocation problem represented by an instance of ESPCPR



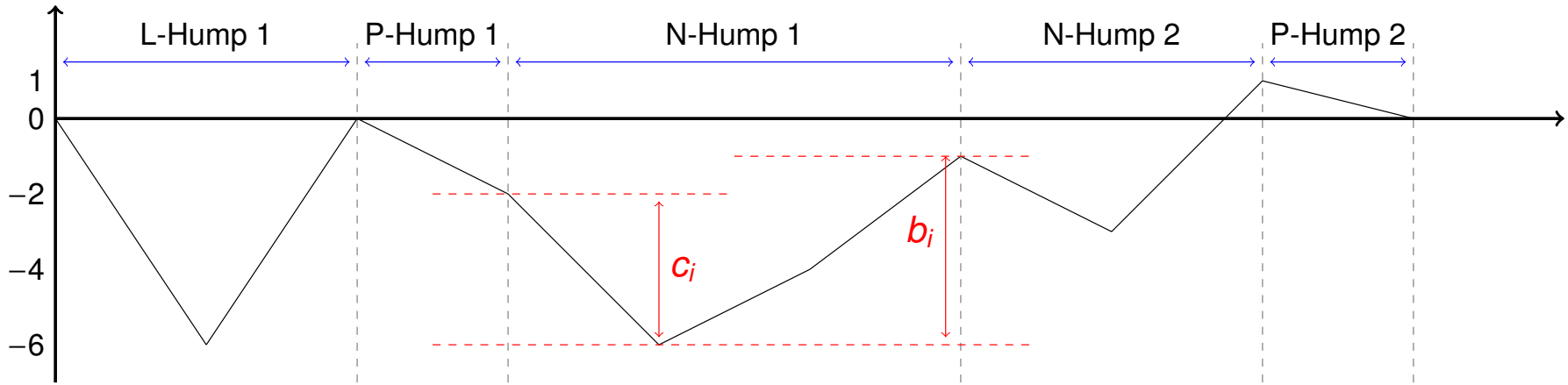
An optimal sequence

The parallel chain case

- Resource-constrained single-machine scheduling problem
- (Abdel-Wahab and Kameda, 1978) proposed an algorithm to minimize maximum cumulative cost
  - Determination of N-humps, L-humps and P-humps
  - N-humps : Set of production sub-chains
  - L-humps : Set of null sub-chains
  - P-humps : Set of consumption sub-chains
  - c_i corresponds to the fall of the hump i
  - b_i corresponds to the rise of the hump i



An instance of parallel chain problem



The sequence (chain 1 – chain 2 – chain 3)

The parallel chain case

- The Johnson's rule can be extended to solve the parallel chain case
  - The jobs correspond to the humps
  - c_i (the fall of the hump i) corresponds to the processing time on the first machine
  - b_i (the rise of the hump i) corresponds to the processing time on the second machine
  - First sequence the N-humps of the chains in non-decreasing order of their falls
  - To the partial sequence attach the L-humps followed by P-humps in non-increasing order of their rises

- We can extend this algorithm to ESPCPR with parallel chain precedence constraints
- **(Abdel-Wahab and Kameda, 1978)** proposed an algorithm to solve the series-parallel case.
  - The precedence relation is represented by an SP graph
  - Find two parallel chains
  - Apply the Parallel Chain Algorithm
  - Replace the two chains by the resulting single chain and iterate

Project Scheduling with inventory constraints

(storage resources)

- Inventory constraints refer to non-renewable resources (storage resources)
- The level of resource stock must always respect a minimum and a maximum value
- The problem consists in computing a schedule that is resource-feasible, time-feasible and that minimizes makespan
- Neumann and Schwindt proposed a branch and bound method to solve the problem and truncated it to a filtered beam search heuristic
- Laborie proposed a constraint programming method for solving the problem
- We can model a project scheduling problem with inventory constraints by ESPCPR
  - Replace resource k by two new resources denoted k_1 and k_2.
  - If event i produces a quantity r_{ik}, it will produce a quantity $a_{ik_1} = r_{ik}$ of resource k_1 and consume a quantity $a_{ik_2} = -r_{ik}$ of resource k_2

Plan

- Introduction
- ESPCPR
- Literature
- MILP for ESPCRP
- Strict Order Algorithm
- Classical problems
- Lower bounds
- Results
- Conclusion

MILP FORMULATIONS FOR THE ESPCPR

- **Difficulty:**
  - Combination of precedence and resource constraints
- **Option 1:** using time-indexed variables
  - x_{it}**:** *i* event, *t* time
  - **Drawback:** Pseudo-polynomial number of constraints and variables
- **Option 2:** using sequencing variables
  - y_{ij}**:** *i* event, *j* event
  - **Drawback:** Relaxation of poor quality

MILP FORMULATIONS FOR THE ESPCPR

- Discrete-time formulation (DT)
  - **Variables:** Pseudo-polynomial number of variables
  - **Constraints:** Pseudo-polynomial number of constraints
- Disaggregated discrete-time formulation (DDT)
  - **Variables:** Pseudo-polynomial number of variables
  - **Constraints:** Pseudo-polynomial number of constraints
- Flow-based continuous-time formulation (FCT)
  - **Variables:** Polynomial number of variables $O(n^2)$
  - **Constraints:** Polynomial number of constraints $O(n^3)$
- Event partitioning based formulation (EP)
  - **Variables:** Polynomial number of variables $O(n^2)$
  - **Constraints:** Polynomial number of constraints $O(n^2)$

Discrete-time formulation (DT)

- DT was initially introduced for the RCPSP by (Pritsker et al., 1969)
- It was adapted for the RCPSP/CPR (RCPSP with Consumption and Production of resources) by (Koné et al., 2013)
- One constraint for each precedence relation
- x_{it}: binary variable indexed by both events and time
  - $x_{it} = 1$ if event i occurs at time t
  - $x_{it} = 0$ otherwise

Discrete-time formulation (DT)

Minimize
$$\sum_{t=ES_{n+1}}^{LS_{n+1}} tx_{n+1,t} \tag{1}$$

subject to
$$\sum_{t=ES_i}^{LS_i} tx_{it} + v_{ij} \leq \sum_{t=ES_j}^{LS_j} tx_{jt} \qquad \forall (i,j) \in U \tag{2}$$

$$\sum_{\tau=0}^{t} \sum_{i=0}^{n+1} a_i^k x_{i\tau} \geq 0 \qquad \forall k \in K, \forall t \in H \tag{3}$$

$$\sum_{t=ES_i}^{t=LS_i} x_{it} = 1 \qquad \forall i \in X \tag{4}$$

$$x_{it} = 0 \qquad \forall i \in X, \forall t \in H \setminus \{ES_i, ..., LS_i\} \tag{5}$$

$$x_{it} \in \{0,1\} \qquad \forall i \in X, \forall t \in \{ES_i, ..., LS_i\} \tag{6}$$

Disaggregated discrete-time formulation (DDT)

- Introduced for the RCPSP by (Christofides et al., 1987)
- Adapted for the RCPSP/CPR by (Koné et al., 2013)
- One constraint for each precedence relation and for every time of the scheduling horizon

The model is reinforced by disaggregation of the precedence constraints, i.e. replacing precedence constraints by

$$\sum_{t=\tau}^{LS_i} x_{it} + \sum_{t=ES_j}^{\min\{LS_j,\tau+v_{ij}-1\}} x_{jt} \leq 1 \quad \forall (i,j) \in U, \forall \tau \in [ES_i, LS_i] \qquad (7)$$

Flow-based continuous-time formulation (FCT)

- Initially introduced for the RCPSP by (Artigues et al., 2003)
- Adapted for the RCPSP/CPR by (Koné et al., 2013)
- Decision variables:
  - y_{ij}: sequential binary variable needed for each pair of events (i,j) to determine whether event j is processed after event i
  - S_i: continuous time variable required for each event i to determine its occurrence time
  - f_{ijk}: continuous flow variable required to indicate the quantity of resource k transferred from event i to event j

Flow-based continuous-time formulation (FCT)

Minimize S_{n+1} (8)

Precedence constraints:

$$1 \leq y_{ij} + y_{ji} \leq 2 \quad \forall (i,j) \in X^2, i < j \quad (9)$$

$$y_{il} + 1 \geq y_{ij} + y_{jl} \quad \forall (i,j,l) \in X^3 \quad (10)$$

$$S_j - S_i \geq v_{ij} \quad \forall (i,j) \in U \quad (11)$$

$$S_j - S_i \geq M_{ij}(y_{ij} - 1) \quad \forall (i,j) \in X^2 \quad (12)$$

Resource constraints:

$$f_{ijk} \leq \min(a_i^k, |a_j^k|) y_{ij} \quad \forall k \in K, \forall (i,j) \in P_k \times C_k \quad (13)$$

$$\sum_{i \in P_k} f_{ijk} = |a_j^k| \quad \forall k \in K, \forall j \in C_k \quad (14)$$

$$\sum_{j \in C_k} f_{ijk} \leq a_i^k \quad \forall k \in K, \forall i \in P_k \quad (15)$$

Flow-based continuous-time formulation (FCT)

Valid constraints:

$$y_{ij} = 1 \qquad \forall (i,j) \in U, v_{ij} \geq 0 \qquad (16)$$

$$y_{ji} = 0 \qquad \forall (i,j) \in U, v_{ij} > 0 \qquad (17)$$

$$f_{ijk} \geq 0 \qquad \forall k \in K, \forall (i,j) \in P_k \times C_k \qquad (18)$$

$$y_{ij} \in \{0,1\} \qquad \forall (i,j) \in X^2 \qquad (19)$$

$$ES_i \leq S_i \leq LS_i \qquad \forall i \in X \qquad (20)$$

Event partitioning based formulation (EP)

Formulation based on variables indexed by event subsets

- The idea is to construct feasible schedules by partitioning the set of events into several subsets
- Each subset contains events having the same occurrence time

Let $\Phi = \{\phi_0, \phi_1, ..., \phi_{n+1}\}$ be a partition of X

- z_{ie}: binary variable equal to 1 if event i belongs to subset ϕ_e, and to 0 otherwise
- S_i: continuous time variable required for each event i to determine its occurrence time
- t_e: the occurrence time of each event included in ϕ_e
- s_{ek}: continuous variable required to determine the availability of resource k after the execution of each event of ϕ_e

Event partitioning based formulation (EP)

Minimize

$$t_{n+1} \tag{21}$$

Precedence constraints:

$$t_e \geq S_i - M(1 - z_{ie}) \qquad \forall i \in X, \forall e \in [0...n+1] \tag{22}$$

$$S_i \geq t_e - M(1 - z_{ie}) \qquad \forall i \in X, \forall e \in [0...n+1] \tag{23}$$

$$t_{e+1} \geq t_e \qquad \forall e \in [0...n] \tag{24}$$

$$\sum_{e=0}^{n+1} z_{ie} = 1 \qquad \forall i \in X \tag{25}$$

$$S_i + v_{ij} \leq S_j \qquad \forall (i,j) \in U \tag{26}$$

Resource constraints:

$$s_{0k} = \sum_{i=0}^{n+1} a_i^k z_{i0} \qquad \forall k \in K \tag{27}$$

$$s_{ek} = s_{e-1,k} + \sum_{i=1}^{n+1} a_i^k z_{ie} \qquad \forall k \in K, \forall e \in [1...n+1] \tag{28}$$

Event partitioning based formulation (EP)

Valid constraints:

$$ES_i \leq S_i \leq LS_i \qquad \forall i \in X \qquad (29)$$

$$t_e \geq 0 \qquad \forall e \in [1...n+1] \qquad (30)$$

$$t_0 = 0 \qquad (31)$$

$$t_e \geq ES_i z_{ie} \qquad \forall i \in X, \forall e \in [0...n+1] \qquad (32)$$

$$t_e \leq LS_i z_{ie} + LS_{n+1}(1 - z_{ie}) \qquad \forall i \in X, \forall e \in [0...n+1] \qquad (33)$$

$$ES_{n+1} \leq t_{n+1} \leq LS_{n+1} \qquad (34)$$

$$z_{ie} \in \{0,1\} \qquad \forall i \in X, \forall e \in [0...n+1] \qquad (35)$$

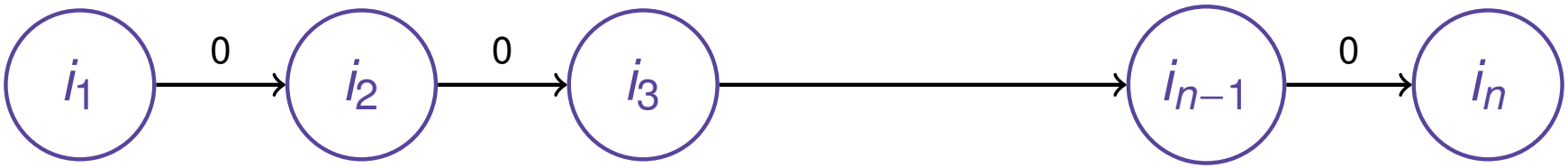
$$s_{ek} \geq 0 \qquad \forall e \in [0...n+1], k \in K \qquad (36)$$

Plan

- Introduction
- ESPCPR
- Literature
- MILP for ESPCRP
- Strict Order Algorithm
- Classical problems
- Lower bounds
- Results
- Conclusion

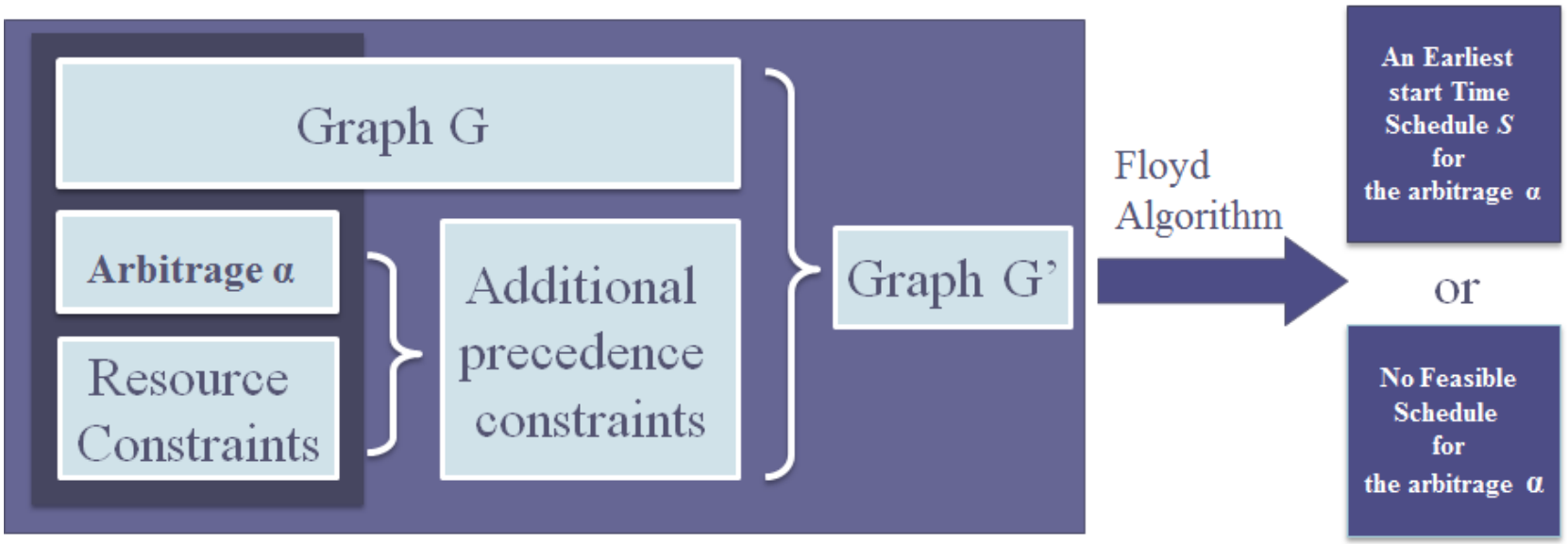
Strict Order Algorithm

- **Arbitrage (Carlier 1984):** An Arbitrage of the resource is a set of conjunctive arcs $\alpha = \{(i_1, i_2), ..., (i_{n-1}, i_n)\}$ with null valuations while $\pi = (i_1, i_2, ..., i_n)$ is a permutation on the set of activities requiring the resource.



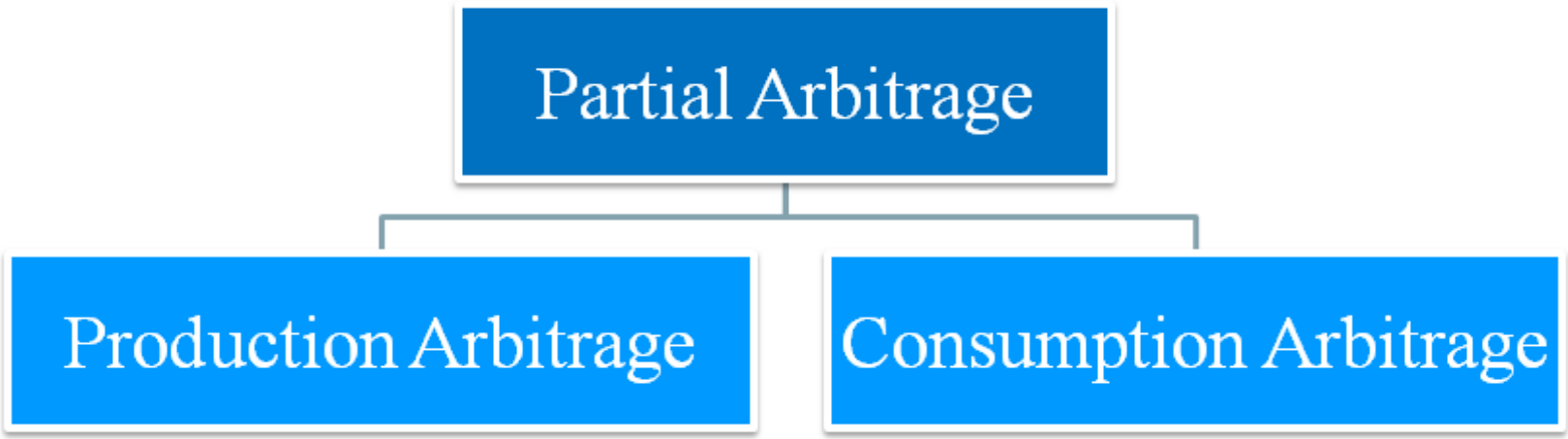
Strict Order Algorithm

- **Complete Arbitrage Method**
  - The idea is to convert all resource constraints into potential constraints and then to use classical algorithms to solve the problem
  - Dominant Schedules
- **Complexity of algorithm of complete arbitrage:** $O(n^3)$.



Strict Order Algorithm

But if we use the Algorithm of Complete Arbitrage for enumeration, it will be too costly. So an idea is to limit the arbitrage to production activities or consumption activities.



Strict Order Algorithm

Arbitrage of Consumption activities

Case of positive arc weights

- Polynomial algorithm $O(n^2)$
- For each iteration : we schedule production events as early as possible. Then we start consumption events without unscheduled direct predecessors as early as possible.

Case of non-negative arc weights

- Polynomial algorithm $O(n^3)$

Case of negative arc weights

- Pseudo-polynomial algorithm

Remark: It can be generalized to a non Strict Order Algorithm

Plan

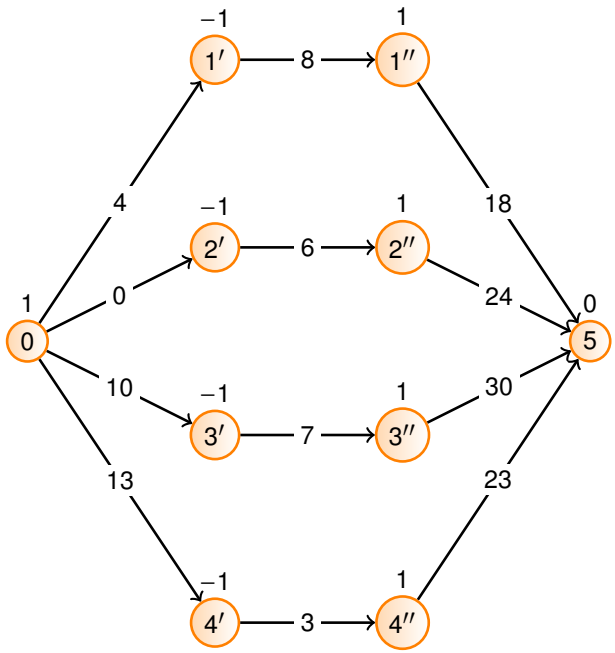
- Introduction
- ESPCPR
- Literature
- MILP for ESPCRP
- Strict Order Algorithm
- Classical problems
- Lower bounds
- Results
- Conclusion

The one-machine scheduling problem and ESPCPR

- **Notation:** $1/r_i, q_i/C_{max}$
- I: Set of n non-preemptive operations
- One available machine
- i: Index of operation
- p_i: Processing time
- r_i: Release date
- q_i: The latency between completion time of operation i and completion time of the project.
- **Aim:** Minimize the makespan

Operation	1	2	3	4
release date	4	0	10	18
processing time	8	6	7	3
tail	18	24	30	23

An instance of the one-machine scheduling problem



ESPCPR

The one-machine scheduling problem and JPS

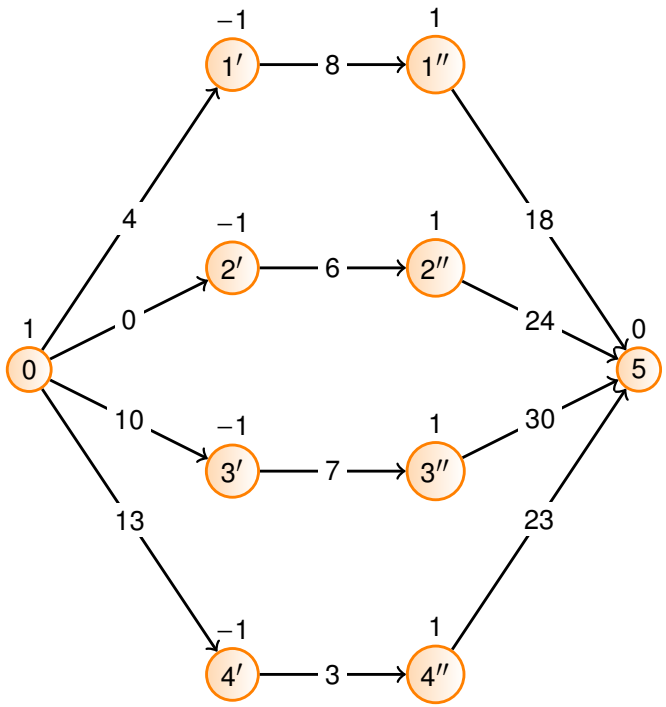
Jackson's Preemptive Schedule

- **Notation:** JPS
- Lower bound for $1/r_i, q_i/C_{max}$ **(Carlier and Pinson 1990)**
- List schedule associated with the Most Work Remaining (MWR) priority dispatching rule
- $LB(JPS)$ can be computed in $O(n\log_2 n)$ time
- $LB(JPS) = \max_{J \subseteq I}(h(J))$ with

$$h(J) = \min_{j \in J} r_j + \sum_{j \in J} p_j + \min_{j \in J} q_j$$

Operation	1	2	3	4
release date	4	0	10	18
processing time	8	6	7	3
tail	18	24	30	23

An instance of the one-machine scheduling problem



ESPCPR

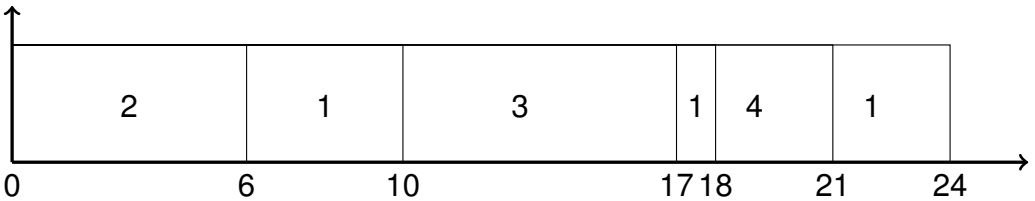
The one-machine scheduling problem and JPS

Jackson's Preemptive Schedule

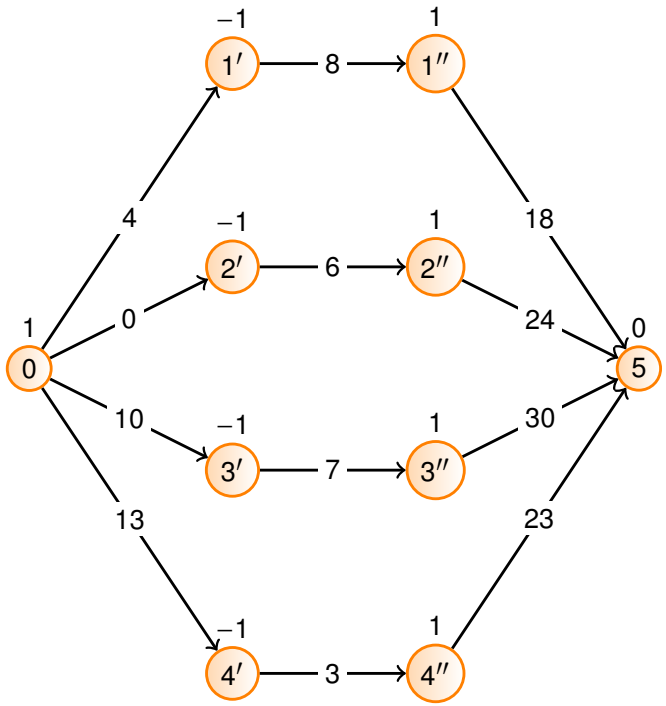
- **Notation:** JPS
- Lower bound for $1/r_i, q_i/C_{max}$ **(Carlier and Pinson 1990)**
- List schedule associated with the Most Work Remaining (MWR) priority dispatching rule

Operation	1	2	3	4
release date	4	0	10	18
processing time	8	6	7	3
tail	18	24	30	23

An instance of the one-machine scheduling problem



Solution of JPS



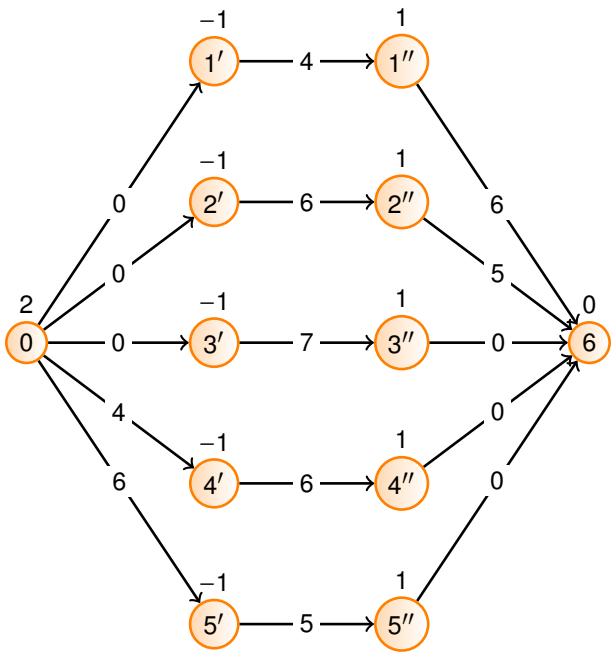
ESPCPR

The m-machine scheduling problem and ESPCPR

- **Notation:** $Pm/r_i, q_i/C_{max}$
- Generalization of $1/r_i, q_i/C_{max}$
- I: Set of n non-preemptive operations
- m available machines
- r_i: Release date
- q_i: The latency between completion time of operation i and completion time of the project.
- **Aim:** Minimize the makespan

Operation	1	2	3	4	5
release date	0	0	0	4	6
processing time	4	6	7	6	5
tail	6	5	0	0	0

An instance of the m-machine scheduling problem where m=2



ESPCPR

The m-machine scheduling problem and JPPS

Jackson's Pseudo Preemptive Schedule

- **Notation:** JPPS
- Lower bound for $Pm/r_i,q_i/C_{max}$ **(Carlier and Pinson 1998)**
- Generalization of JPS
- One processor can be shared by a group of operations
- $LB(JPPS)$ can be computed in $O(n\log n + n.m\log m)$ time

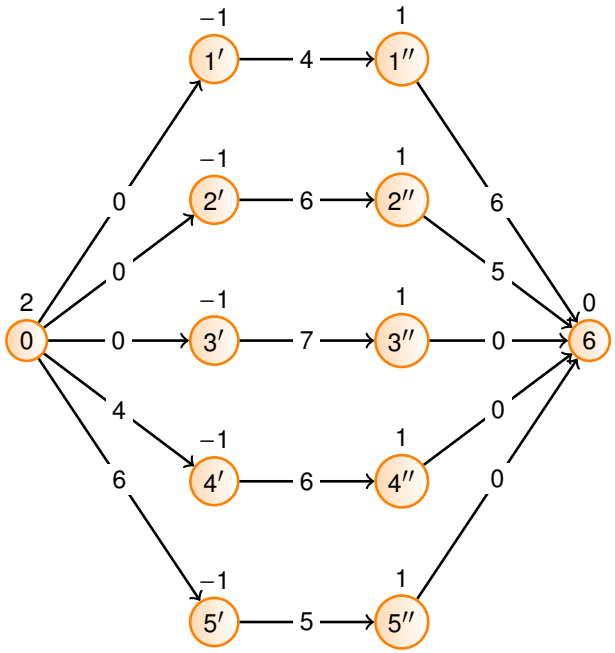
$$LB(JPPS) = \max(\max_{i\in I}(r_i + p_i + q_i), \max_{J\subseteq I,|J|\leq m}(G'(J)))$$

$$G'(J) = \frac{1}{m}(r_{i_1} + r_{i_2} + ...r_{i_m}) + \frac{1}{m}\sum_{j\in J} p_j + \frac{1}{m}(q_{j_1} + q_{j_2} + ...q_{j_m})$$

where $i_1, i_2, ..., i_m$ (resp. $j_1, j_2, ..., j_m$) denote the m first jobs in J rearranged in an ascending order of heads (resp. tails)

Operation	1	2	3	4	5
release date	0	0	0	4	6
processing time	4	6	7	6	5
tail	6	5	0	0	0

An instance of the m-machine scheduling problem where m=2



ESPCPR

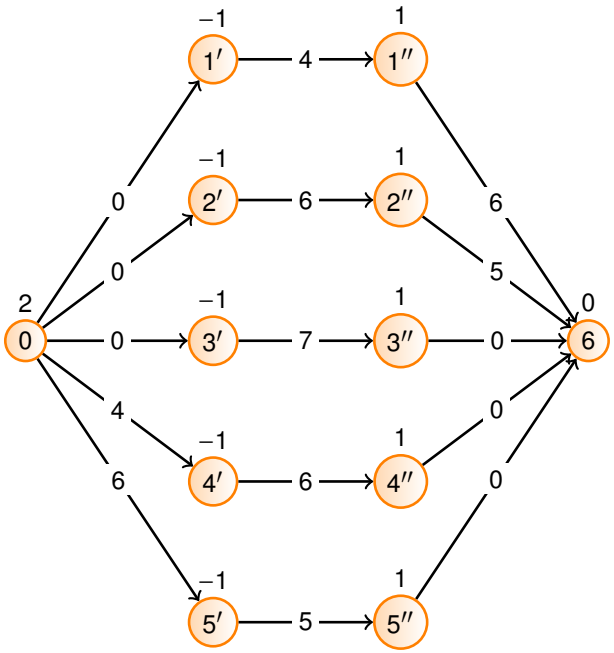
The m-machine scheduling problem and JPPS

Jackson's Pseudo Preemptive Schedule

- **Notation:** JPPS
- Lower bound for $Pm/r_i, q_i/C_{max}$ **(Carlier and Pinson 1998)**
- Generalization of JPS
- One processor can be shared by a group of operations

Operation	1	2	3	4	5
release date	0	0	0	4	6
processing time	4	6	7	6	5
tail	6	5	0	0	0

An instance of the m-machine scheduling problem where m=2



ESPCPR

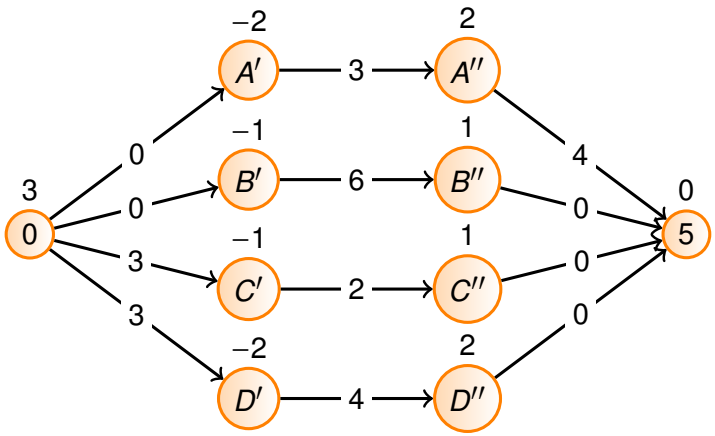
Solution of JPPS

The Cumulative Scheduling Problem and ESPCPR

- **Notation:** CuSP
- Generalization of $Pm/r_i, q_i/C_{max}$
- X: Set of non-preemptive activities
- e_i: The number of resource units required for the operation i
- r_i: Release date
- q_i: Tail
- **Aim:** Minimize the makespan

Operation	A	B	C	D
Release date	0	0	3	3
Processing time	3	6	2	4
Tail	4	0	0	0
Resource	2	1	1	2

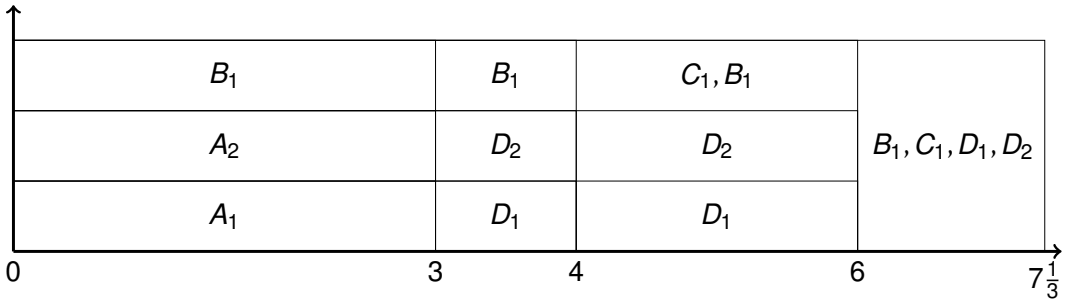
An instance of CuSP(Available resource = 3)



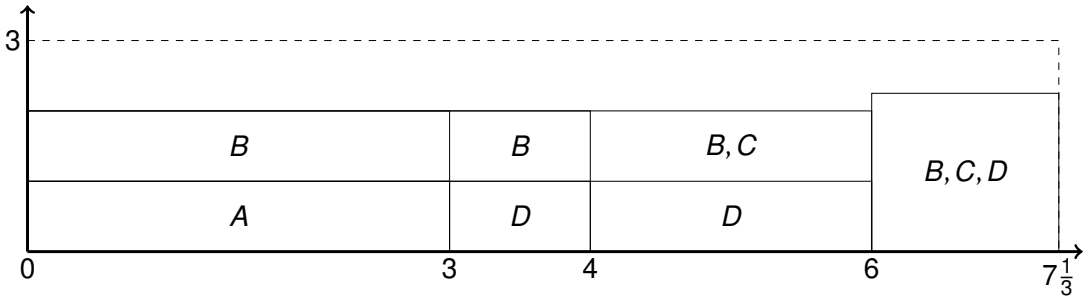
ESPCPR

The Cumulative Scheduling Problem and JPPS

- Two methods to adapt JPPS for CuSP



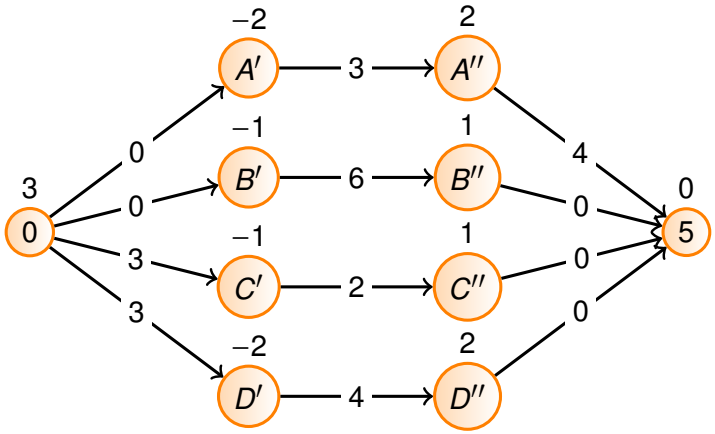
Solution of JPPS with the first strategy



Solution of JPPS with the second strategy

Operation	A	B	C	D
Release date	0	0	3	3
Processing time	3	6	2	4
Tail	4	0	0	0
Resource	2	1	1	2

An instance of CuSP(Available resource = 3)



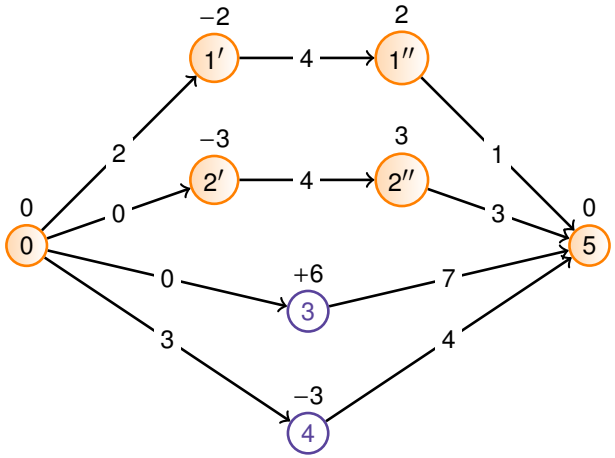
ESPCPR

The Generalized Cumulative Scheduling Problem and ESPCPR

- **Notation:** GCuSP
- Generalization of CuSP
- X: Set of non-preemptive activities
- Y: Set of production and consumption events
- b_i: The number of resource units produced or consumed by event i
  - Positive: production
  - Negative: consumption
- **Aim:** Minimize the makespan

Operation	1	2	3	4
Release date	2	0	0	3
Processing time	4	4	0	0
Tail	1	3	7	4
Resource	2	3	+6	-3

An instance of GCuSP(Available resource = 0)

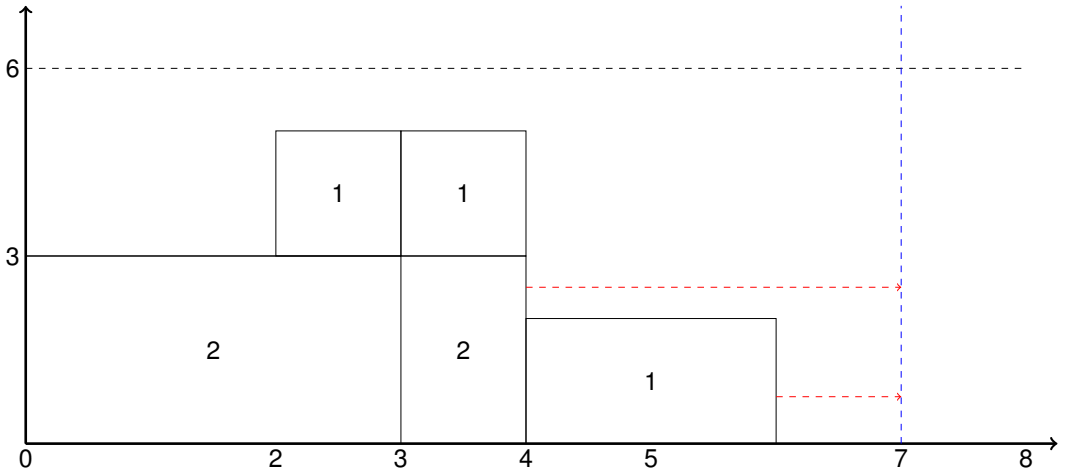


ESPCPR

The Generalized Cumulative Scheduling Problem and JPPS

- **Adaptation of JPPS for GCuSP:** Improved JPPS algorithm

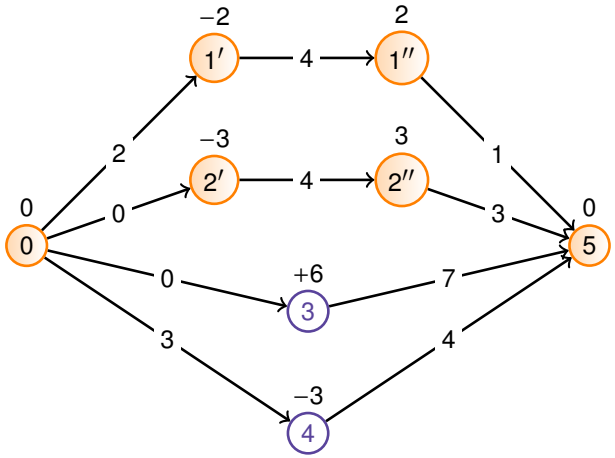
Apply JPPS without considering all consumption events
$C \leftarrow makespan = 7$



Solution of the JPPS without consumption events (makespan = 7)

Operation	1	2	3	4
Release date	2	0	0	3
Processing time	4	4	0	0
Tail	1	3	7	4
Resource	2	3	+6	-3

An instance of GCuSP(Available resource = 0)

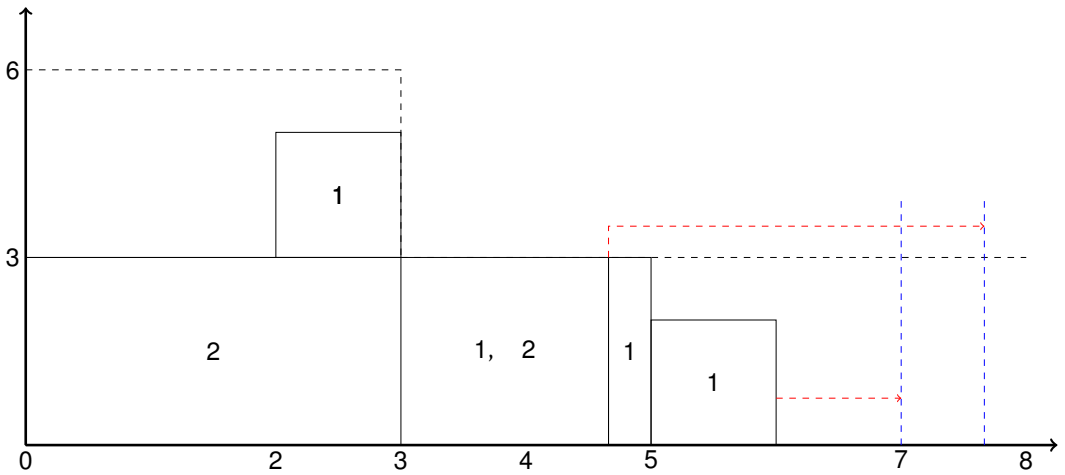


ESPCPR

The Generalized Cumulative Scheduling Problem and JPPS

- **Adaptation of JPPS for GCuSP:** Improved JPPS algorithm

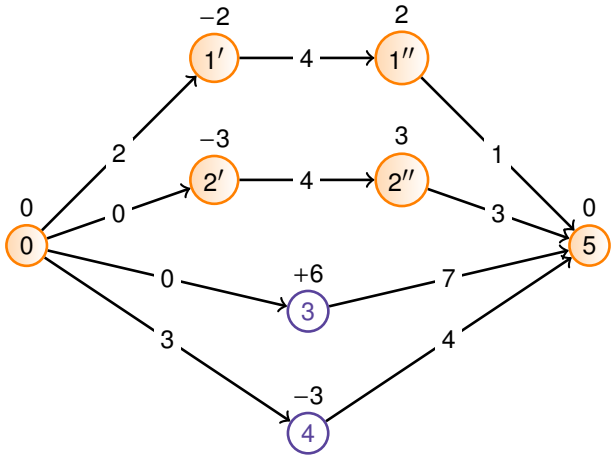
First iteration: $u_4 = 3$, $C < makespan$, $C \leftarrow C + 1$



Solution of the Improved JPPS (makespan = 7,66)

Operation	1	2	3	4
Release date	2	0	0	3
Processing time	4	4	0	0
Tail	1	3	7	4
Resource	2	3	+6	-3

An instance of GCuSP(Available resource = 0)

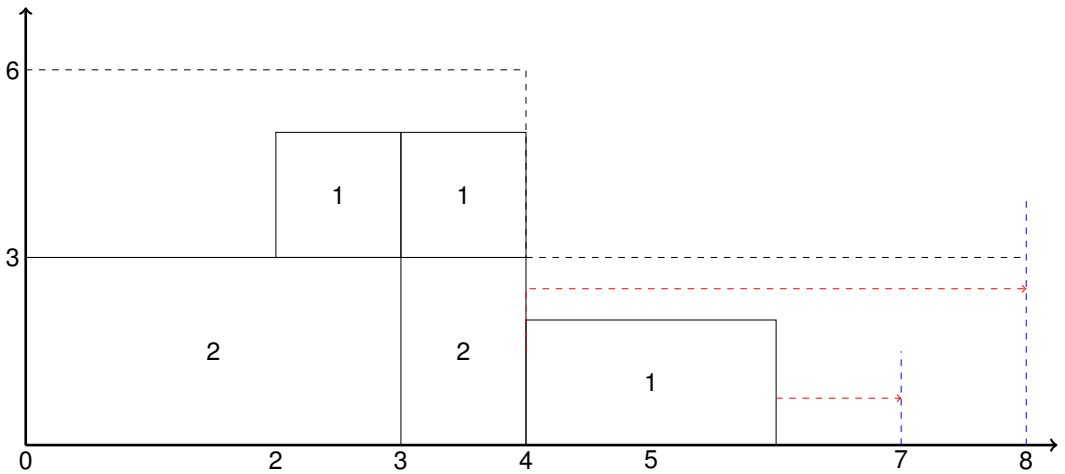


ESPCPR

The Generalized Cumulative Scheduling Problem and JPPS

- **Adaptation of JPPS for GCuSP:** Improved JPPS algorithm

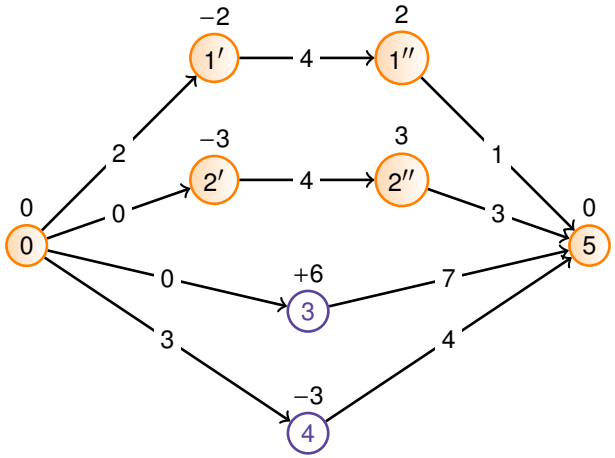
Second iteration: $u_4 = 4$, $C = makespan \Rightarrow Lb = 8$



Solution of the Improved JPPS (makespan = 8)

Operation	1	2	3	4
Release date	2	0	0	3
Processing time	4	4	0	0
Tail	1	3	7	4
Resource	2	3	+6	-3

An instance of GCuSP(Available resource = 0)



ESPCPR

Plan

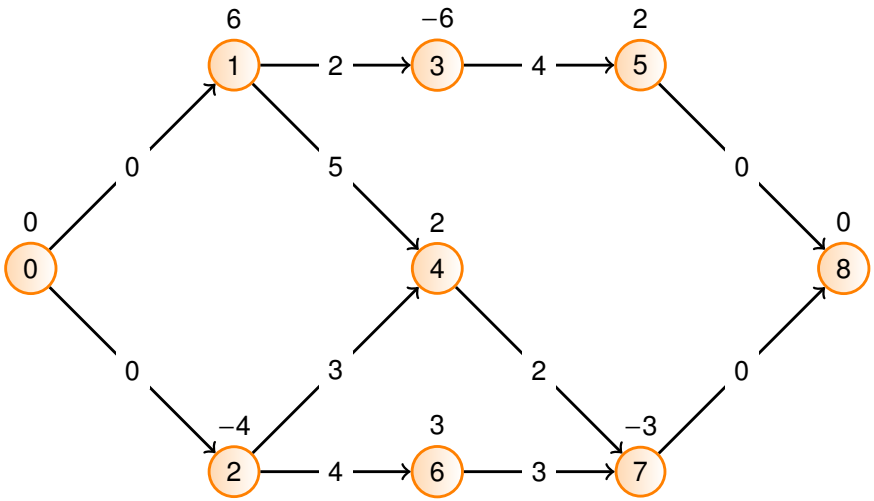
- Introduction
- ESPCPR
- Literature
- MILP for ESPCRP
- Strict Order Algorithm
- Classical problems
- Lower bounds
- Results
- Conclusion

Lower bounds for ESPCPR

- **Adaptation of Jackson's Pseudo-Preemptive Schedule (JPPS)**
  - Initially developed for calculating a lower bound for the Cumulative Scheduling Problem (CuSP).
  - ESPCPR ⇒ Transportation Problem ⇒ GCuSP.
- **Adaptation of the Shifting Algorithm**
  - For solving the project scheduling problem with non-renewable resources.
- **A destructive bound using Network Flows**
  - First set the value of C_{max}.
  - ESPCPR ⇒ Transportation Problem.
  - If the Transportation Problem doesn't admit a solution ⇒ $(C_{max}+1)$ is a lower bound

JPPS and ESPCPR

An instance of ESPCPR

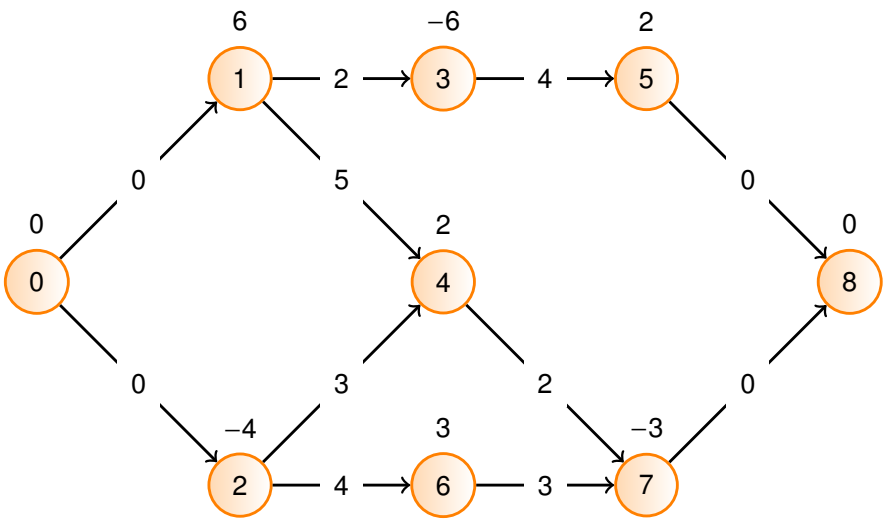


An instance of GCuSP

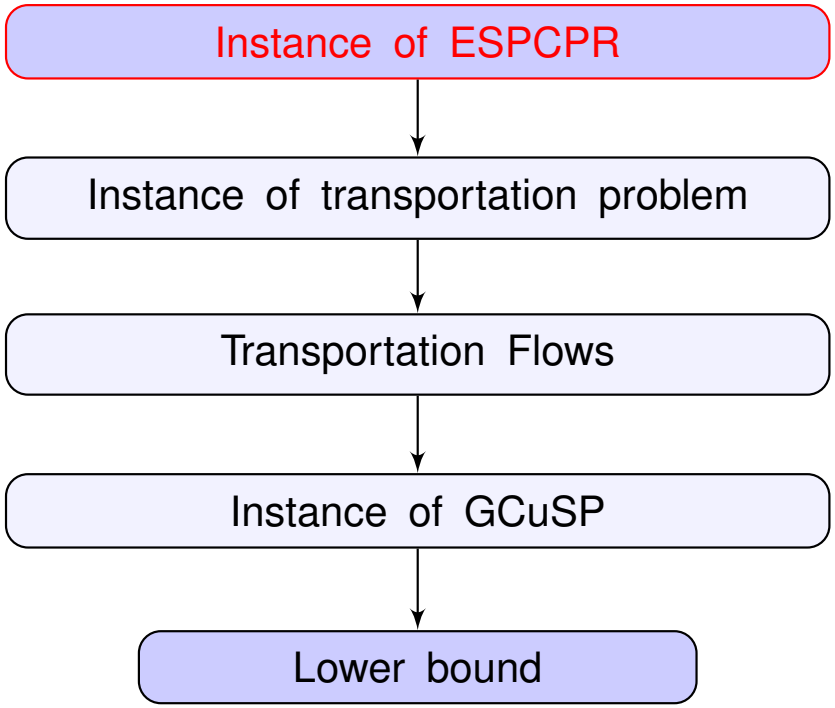
Name	Release date	Tail	Duration	Required resource
activity (2,4)	0	2	3	1
activity (2,6)	0	3	4	3
activity (3,5)	2	0	4	2
event (1)	0	6	0	produces 6
event (4)	5	2	0	produces 1
event (3)	2	4	0	consumes 4
event (7)	7	0	0	consumes 3

JPPS and ESPCPR

- Calculate longest distances between each pair of events i and j, where i = consumption event and j = production event.

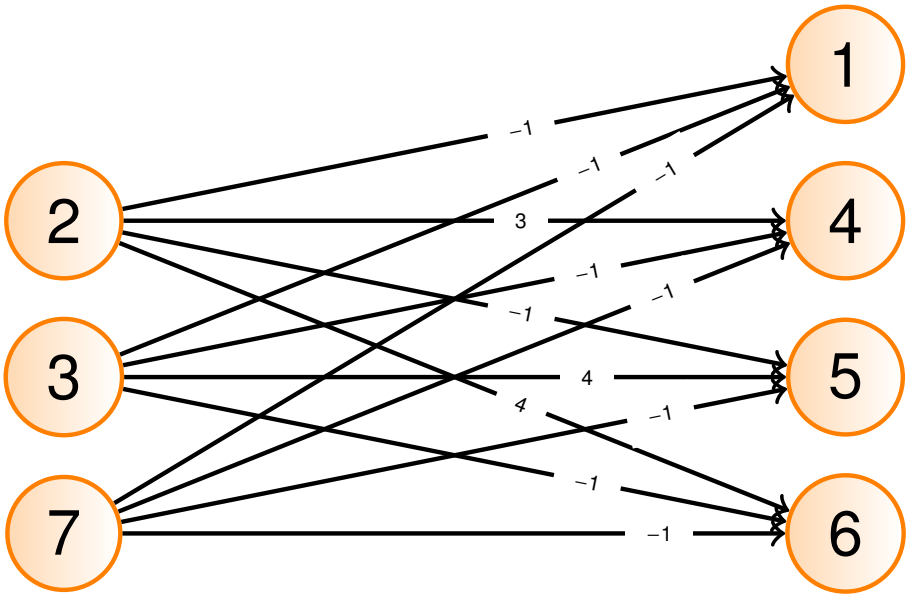


An instance of ESPCPR

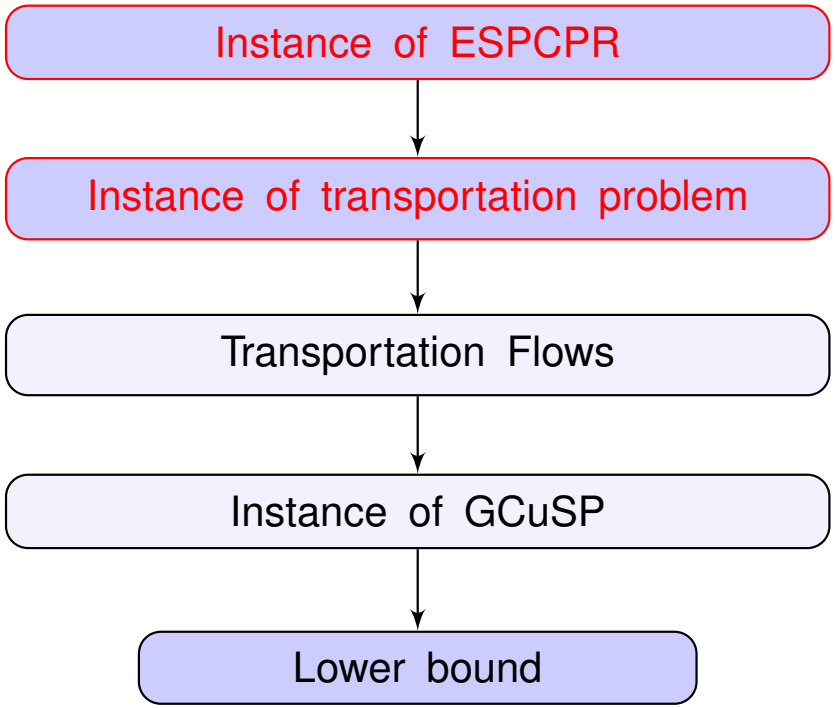


JPPS and ESPCPR

- Applying a heuristic to solve the transportation problem.

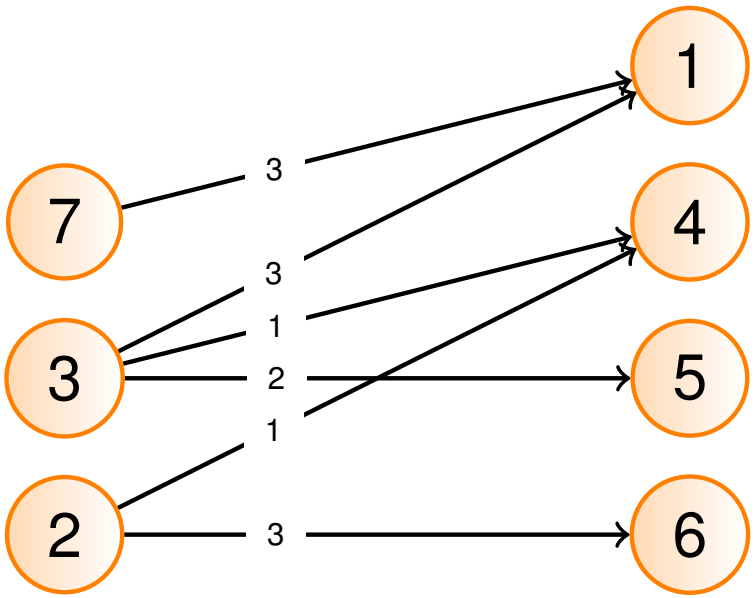


A transportation problem associated with the example

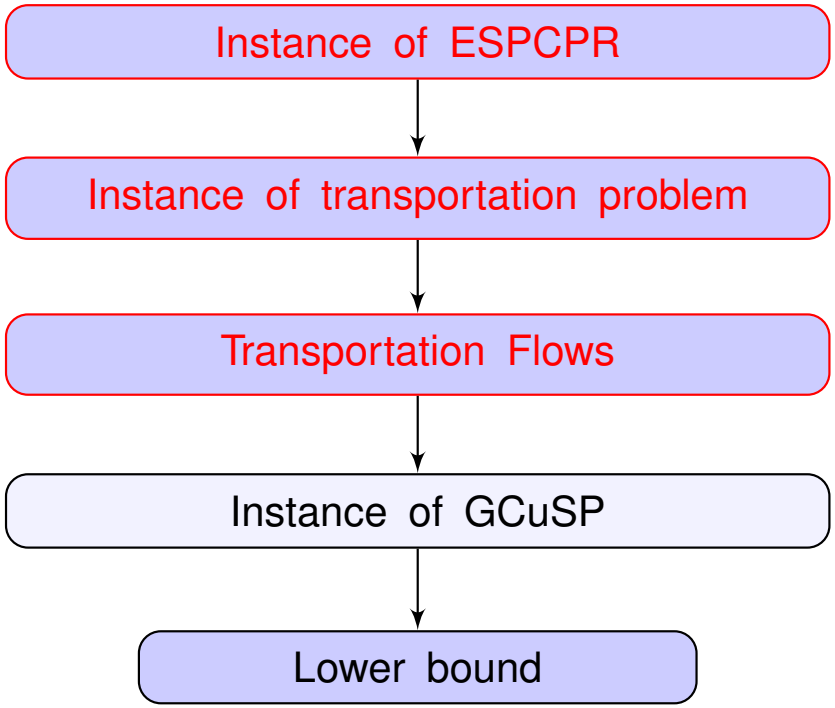


JPPS and ESPCPR

- Consider each flow as an operation of CuSP.



Solution of the transportation problem



JPPS and ESPCPR

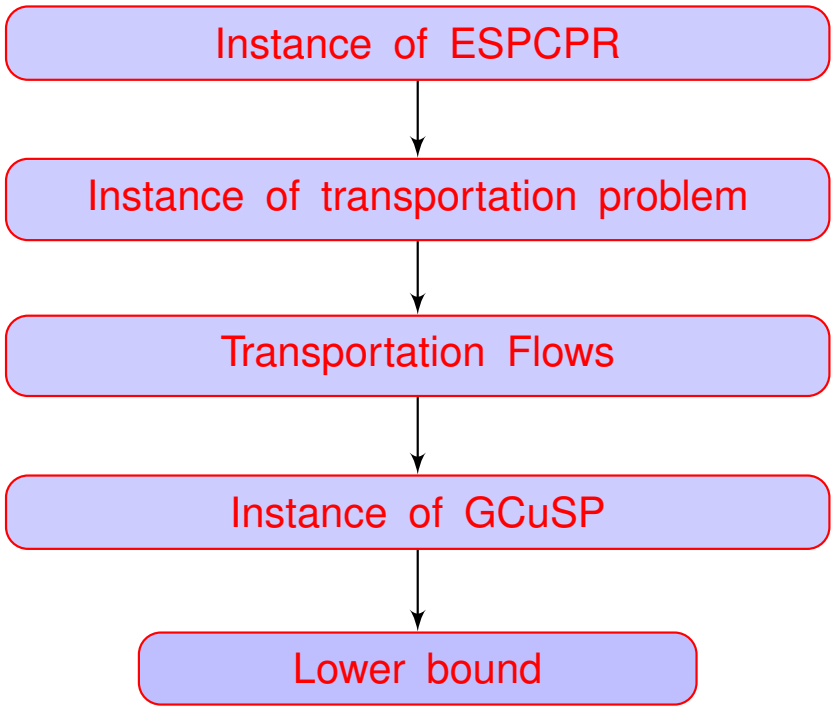
- Applying the improved JPPS.

  The obtained lower bound is equal to 8

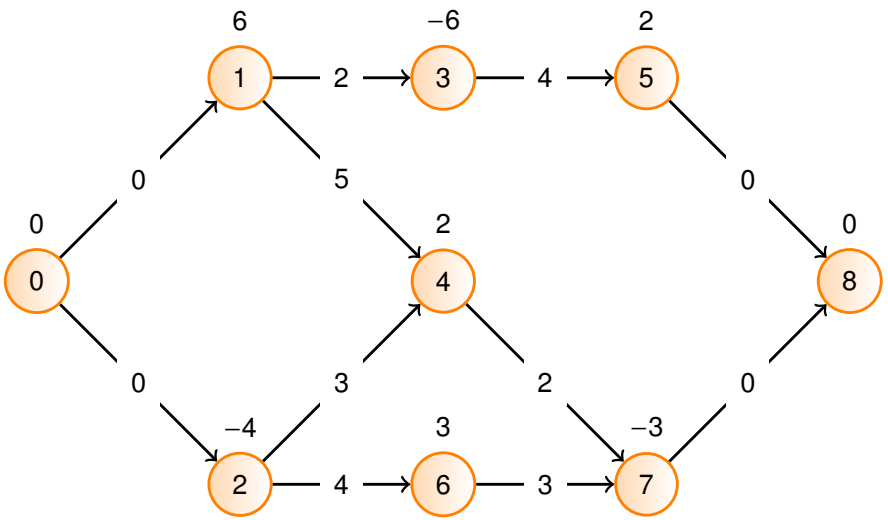
  (Two iterations)

Name	Release date	Tail	Duration	Required resource
activity (2,4)	0	2	3	1
activity (2,6)	0	3	4	3
activity (3,5)	2	0	4	2
event (1)	0	6	0	produces 6
event (4)	5	2	0	produces 1
event (3)	2	4	0	consumes 4
event (7)	7	0	0	consumes 3

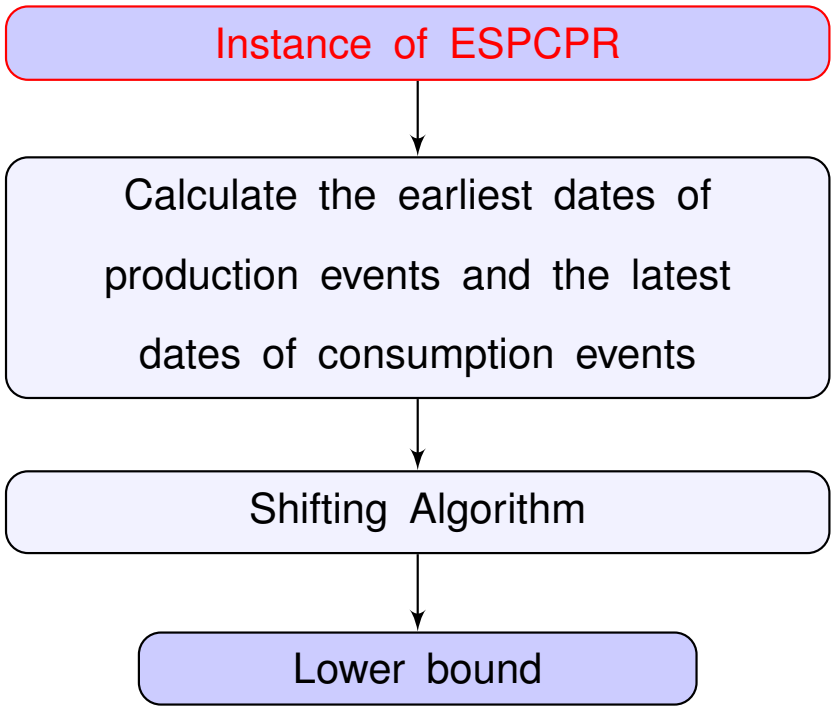
GCuSP instance associated with the example



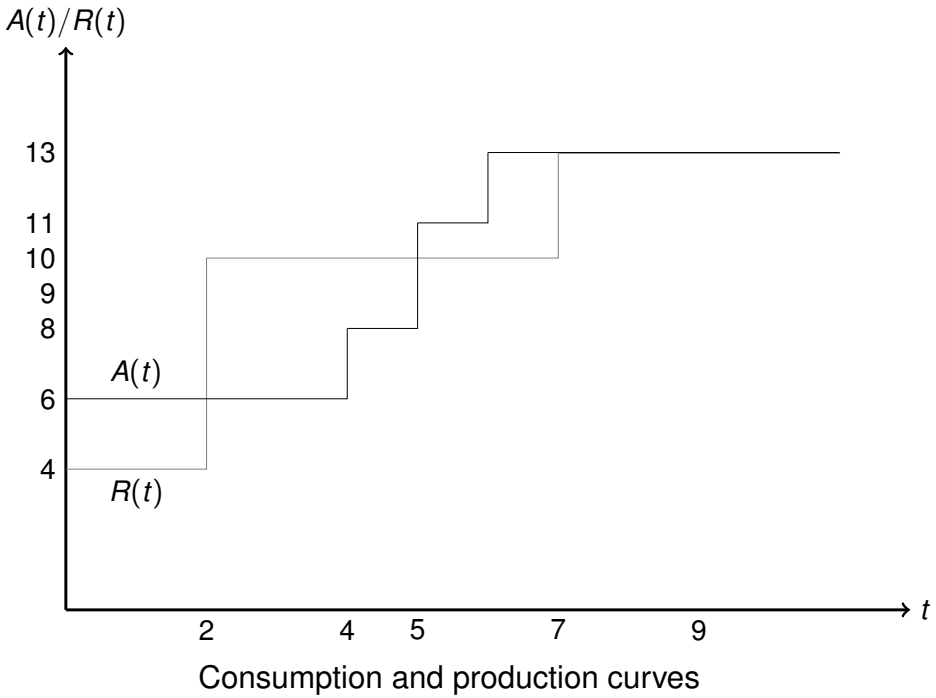
The Shifting Algorithm and ESPCPR



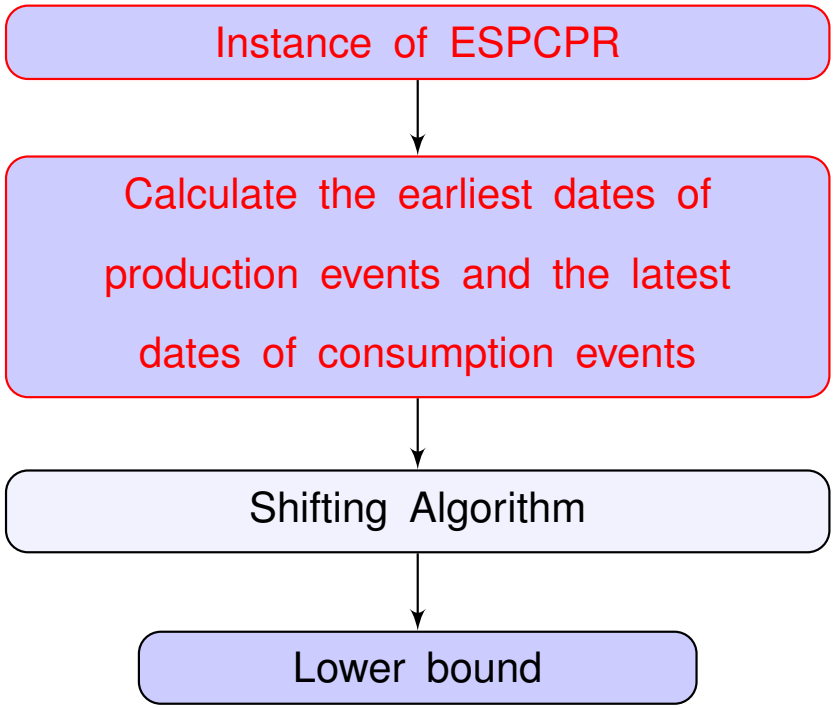
An instance of ESPCPR



The Shifting Algorithm and ESPCPR

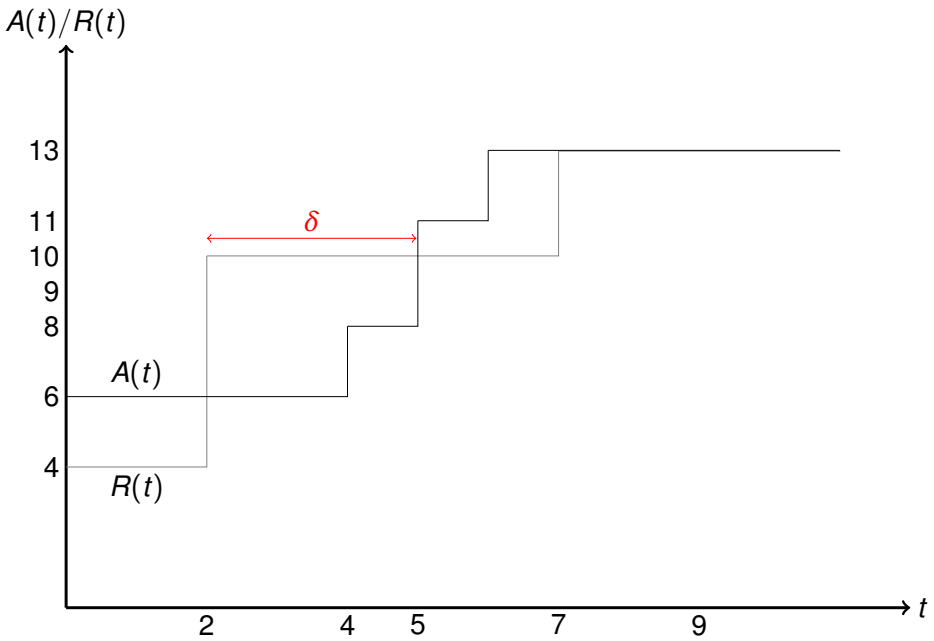


Consumption and production curves

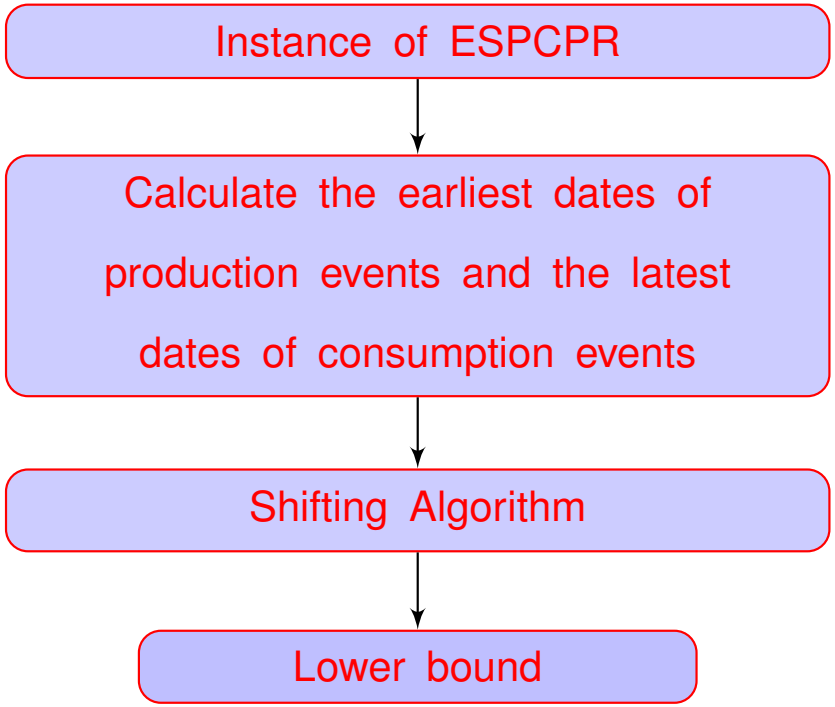


The Shifting Algorithm and ESPCPR

The obtained lower bound is equal to 9

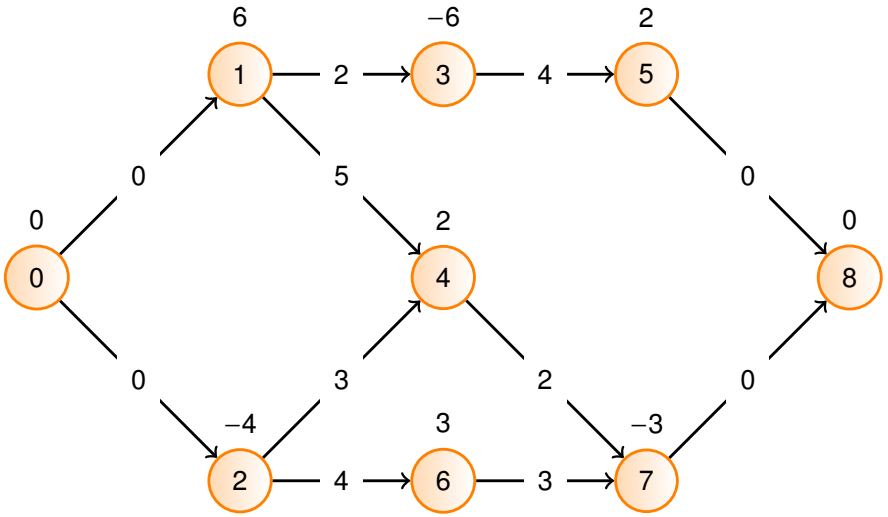


Applying the shifting algorithm

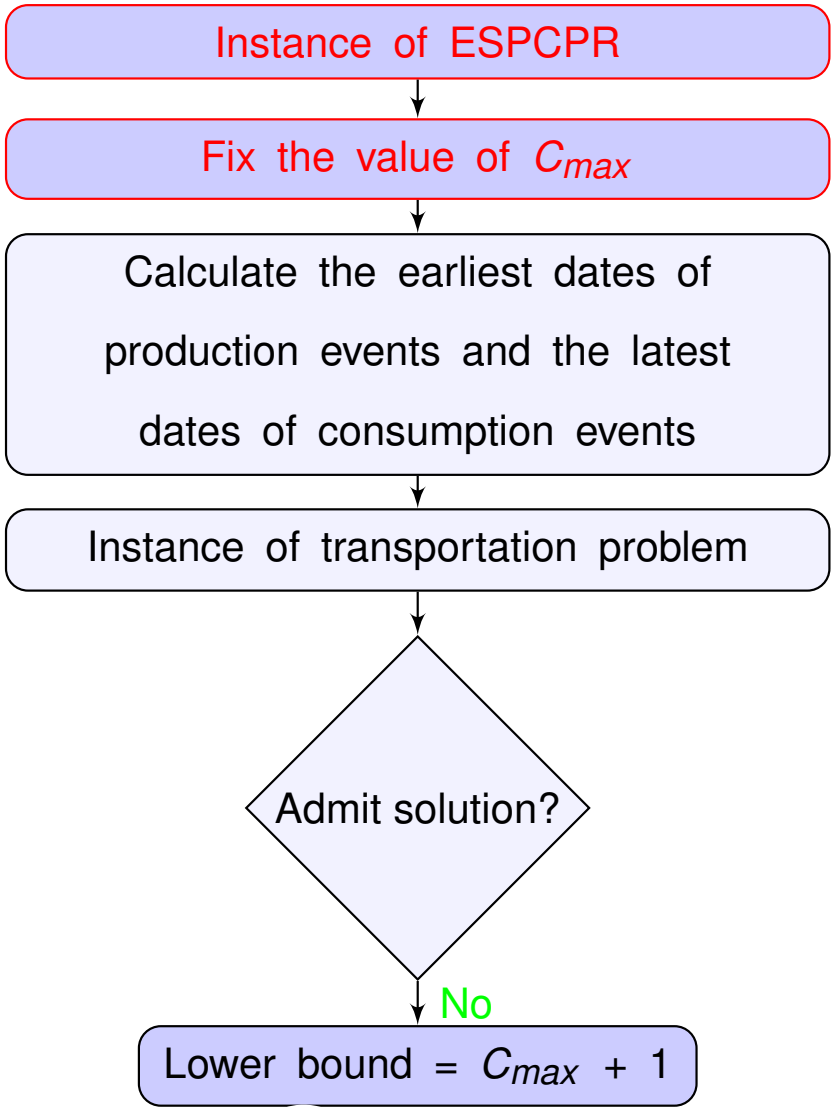


A destructive bound using Network Flows

- $C_{max} = 8$



An instance of ESPCPR

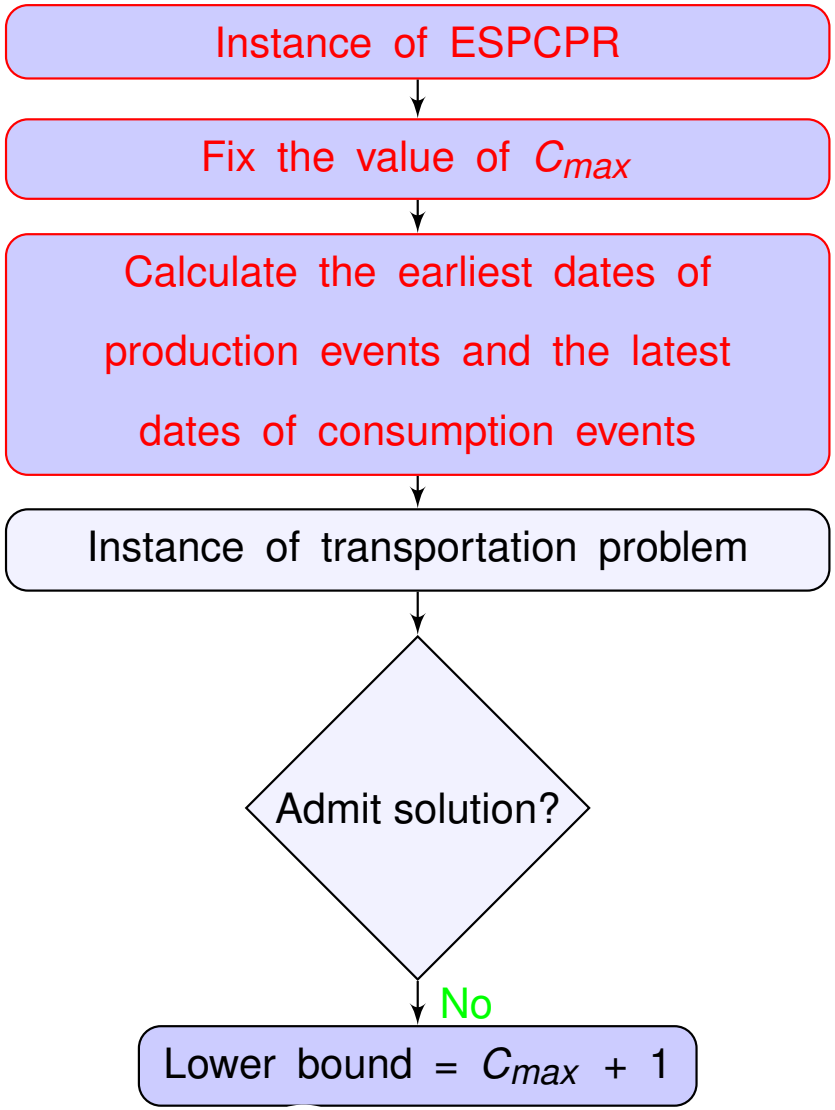


A destructive bound using Network Flows

- $C_{max} = 8$

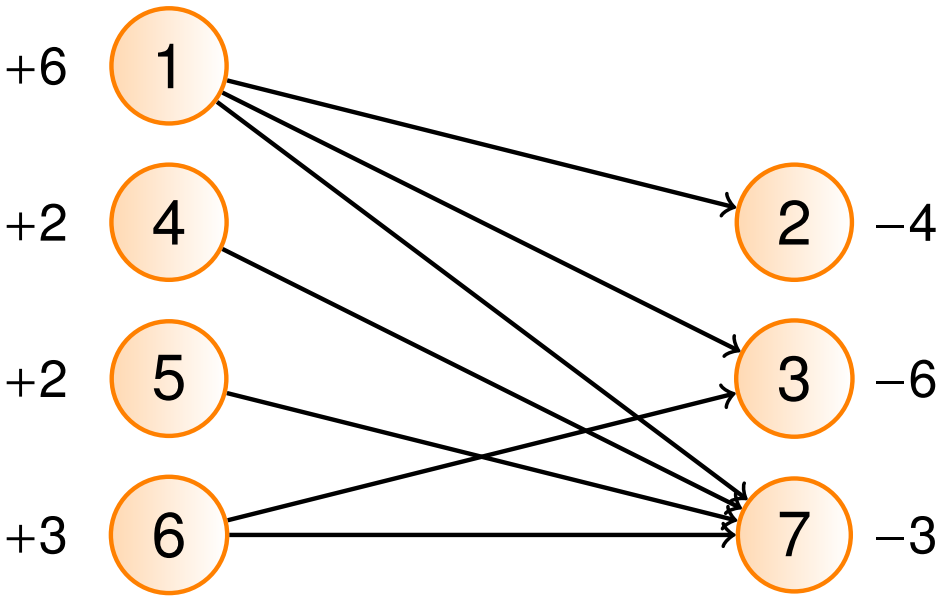
Start time	Event
0	0, 1
1	2
4	3
5	4, 6
6	5
8	7, 8

Start times of all events

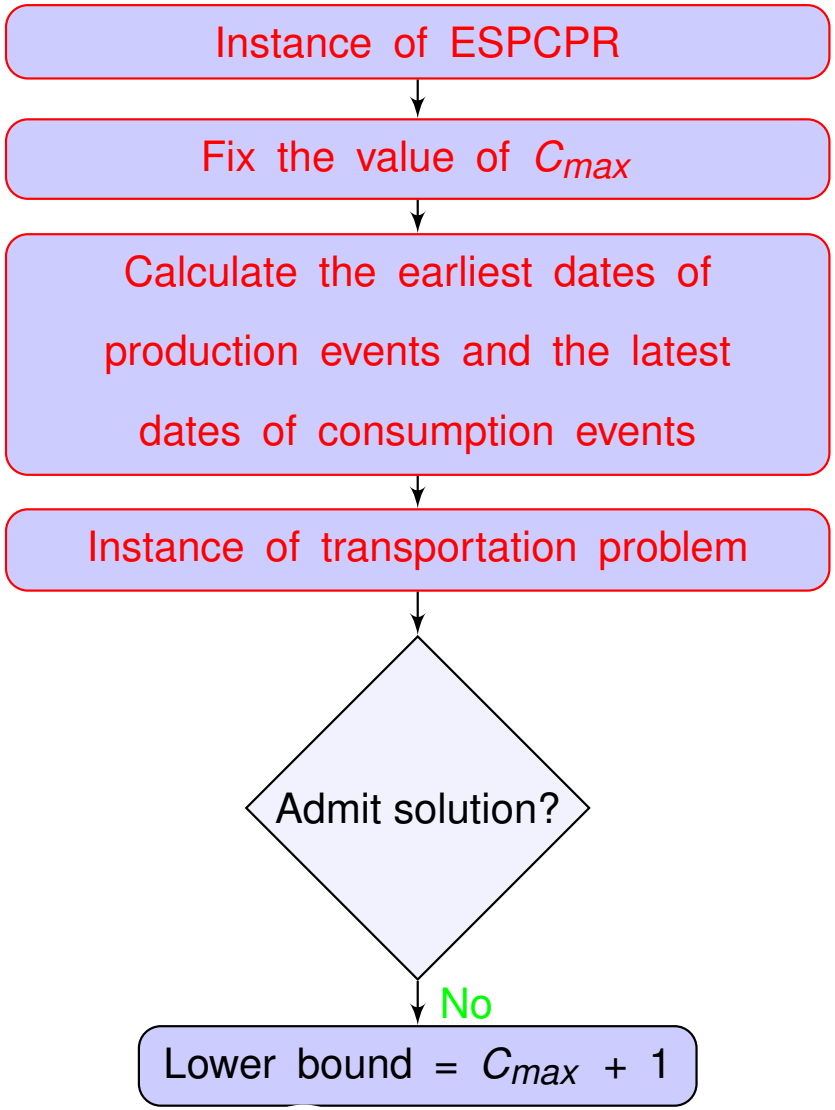


A destructive bound using Network Flows

- $C_{max} = 8$

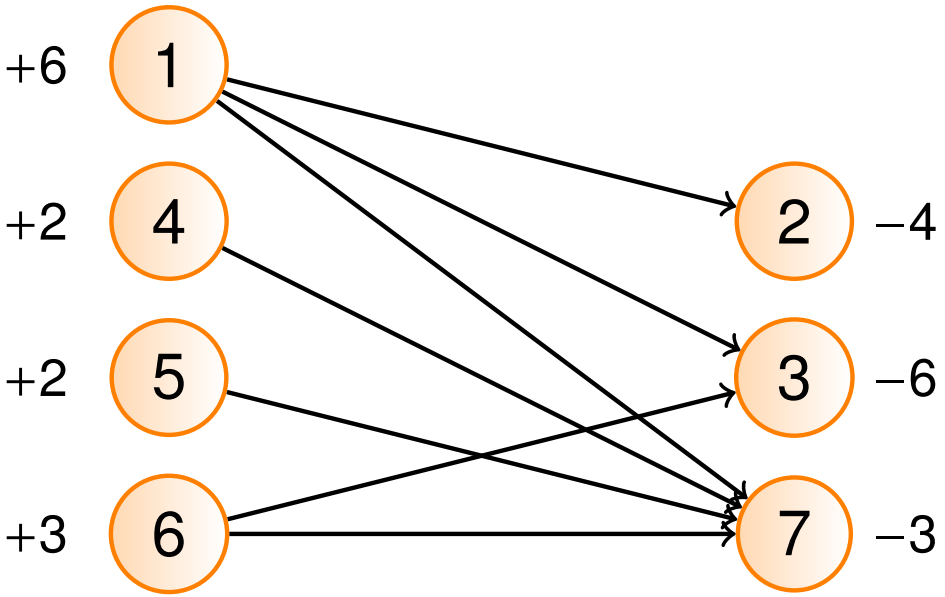


A transportation problem associated with the example

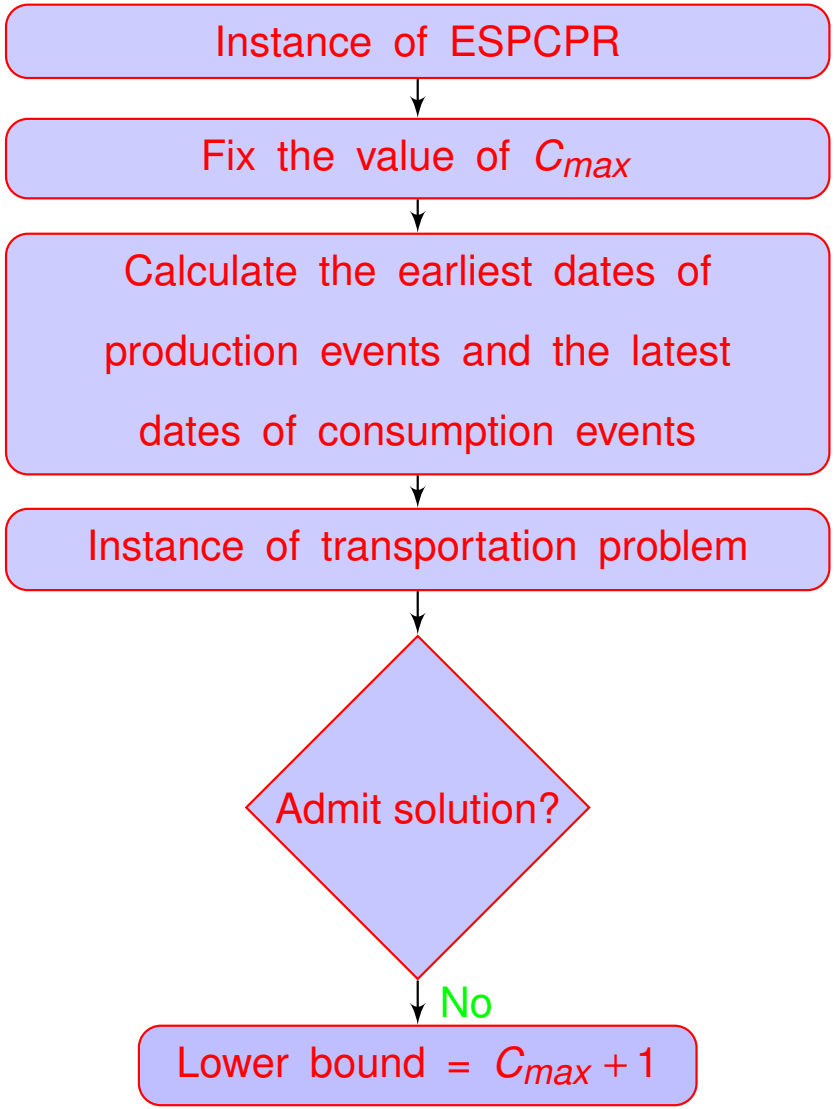


A destructive bound using Network Flows

- The transportation problem doesn't admit a solution ⇒ the lower bound = 9



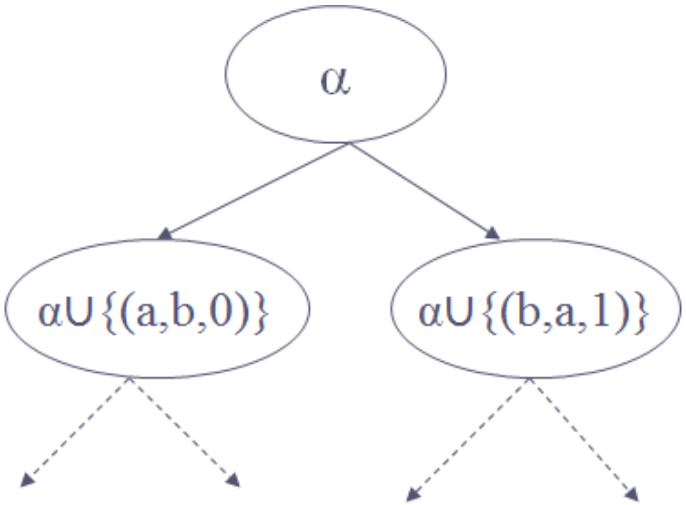
A transportation problem associated with the example



Branch-and-Bound

An idea of branching scheme

- α is a set of arcs which represent precedence constraints between events.
- For a node containing α, we calculate the curve of resource according to the set α.
- *a* is a production event whose occurrence time is smaller than the first resource conflict time *t* and *b* is a consumption event whose occurrence time is larger than *t*.



(a,b,0) is an arc from *a* to *b* valued by 0

Plan

- Introduction
- ESPCPR
- Literature
- MILP for ESPCRP
- Strict Order Algorithm
- Classical problems
- Lower bounds
- Results
- Conclusion

Benchmark of Neumann and Schwindt

- 360 instances with 10, 20, 50, and 100 events
- Min/max delays between events
- 5 resources

Instances	K	Nb_{inst}	Nb_{feas}
NS10	5	90	60
NS20	5	90	43
NS50	5	90	48
NS100	5	90	47

Benchmark details

Results

Data	JPPS			SHIFT		
	Gap	Opt	CPU	Gap	Opt	CPU
NS10	0.41	93.33	0.001	0.50	93.33	**0.000**
NS20	2.34	81.40	0.005	2.13	**83.72**	0.003
NS50	1.57	79.17	0.040	1.54	**81.25**	0.034
NS100	**0.76**	**97.87**	0.263	**0.76**	**97.87**	0.248

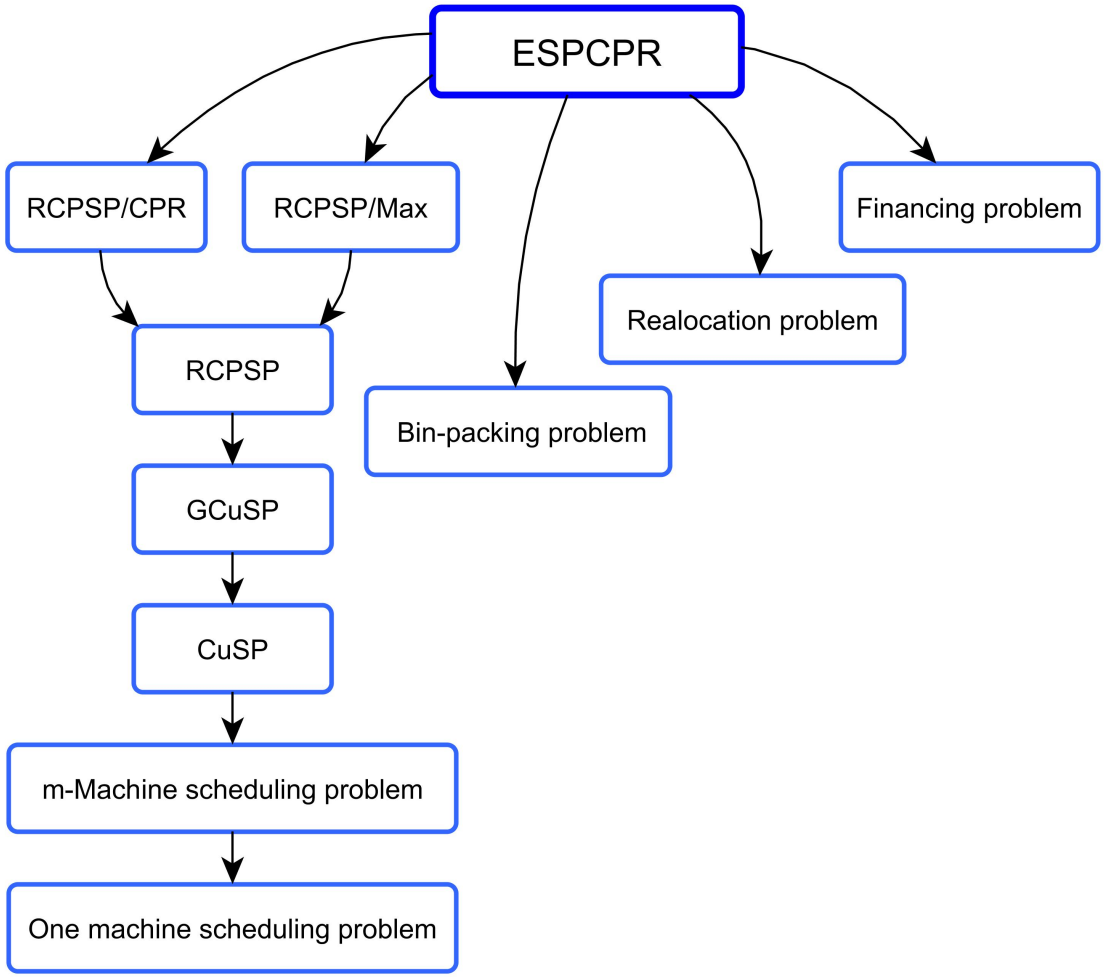
Data	FLOW			LB_0		
	Gap	Opt	CPU	Gap	Opt	CPU
NS10	**0.24**	**98.33**	0.001	2.89	81.67	**0.000**
NS20	**1.93**	**83.72**	0.008	8.86	51.16	**0.002**
NS50	**1.52**	**81.25**	0.077	14.50	34.50	**0,033**
NS100	**0.76**	**97.87**	0.638	18.01	26.00	**0,245**

Computation results

Plan

- Introduction
- ESPCPR
- Literature
- MILP for ESPCRP
- Strict Order Algorithm
- Classical problems
- Lower bounds
- Results
- Conclusion

Conclusion



Model with consumption and production of resources (ESPCPR)

Conclusion

Realization

- Four MILP formulations for ESPCPR
- Relation between ESPCPR and the classical scheduling problems
- Strict Order Algorithm
- Three lower bounds for ESPCPR

Perspectives

- Build a branch and bound method
- Generate new instances of ESPCPR