# Project Management
# Multiple Resource Allocation

Anabela Tereso, Madalena Araújo, Rui Moutinho
anabelat@dps.uminho.pt , mmaraujo@dps.uminho.pt, rumout@gmail.com
Departamento de Produção e Sistemas
Universidade do Minho
4800-058 Guimarães
PORTUGAL

Salah Elmaghraby
elmaghra@eos.ncsu.edu
North Carolina State University
Raleigh, NC 27695-7906
USA

## Introduction – Problem Definition

• Given a multimodal activity network under stochastic conditions, we want to optimize the resource allocation in order to minimize the cost

**Work Content:** $W_r^a \sim \mathrm{Exp}(\lambda_r^a)$

**Resource Allocation:** $0 \leqslant l_r^a \leqslant x_r^a \leqslant u_r^a < \infty$

**Duration:** $Y_a = \max_{r \in \mathrm{R}_a} \left( \frac{W_r^a}{x_r^a} \right)$

**Resource Cost:** $\mathrm{RC}_a = \sum_{r \in \mathrm{R}_a} (x_r^a \times W_r^a)$

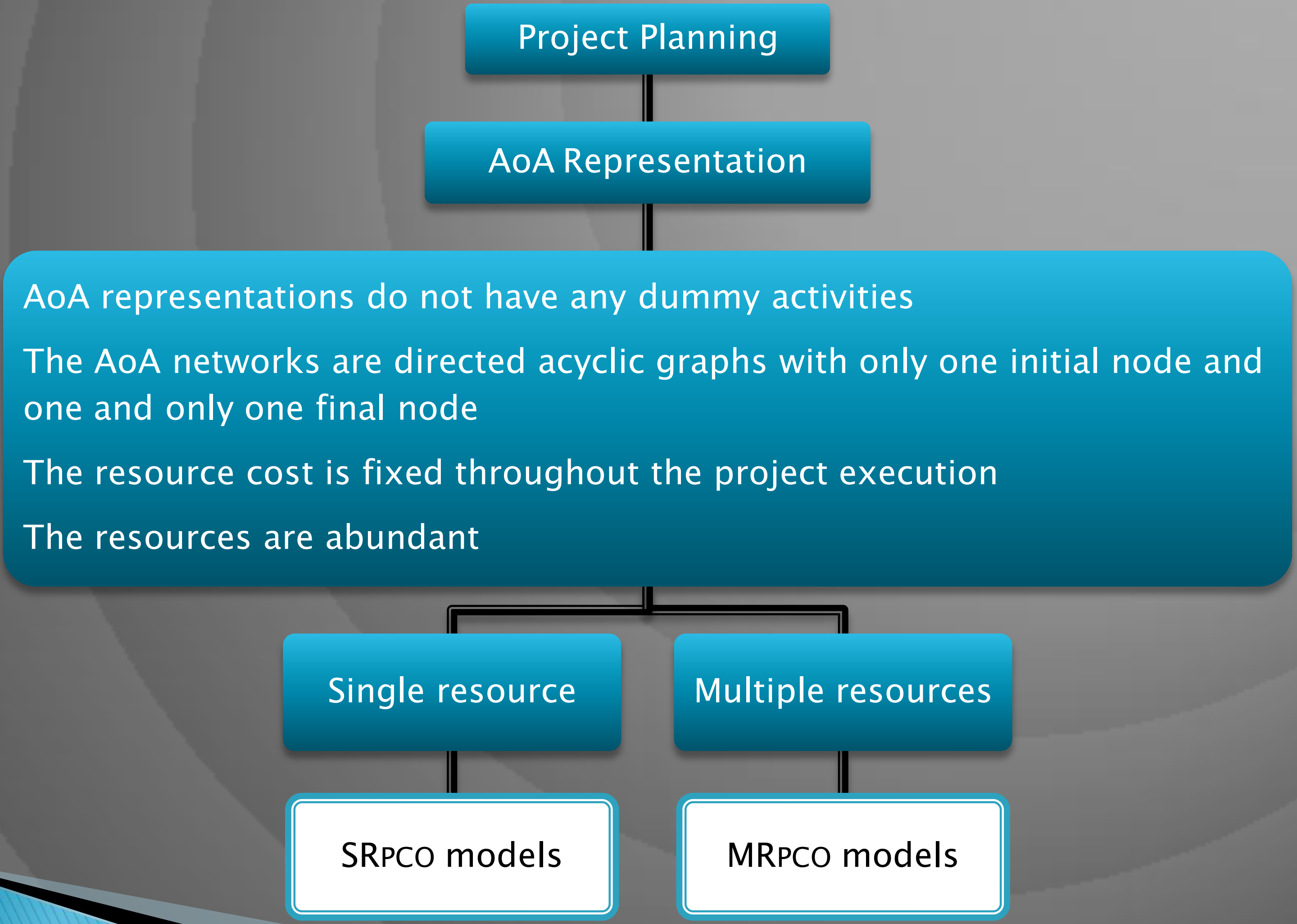
**Due date:** $T$

**Tradiness Cost:** $\mathrm{TC} = c_L \times \max(0, \Upsilon_n - T)$

**One resource / Multiple Resources**

**Goal:** Determine the resource allocation vector $X$, such that the total expected cost is minimized

## Introduction – Review of Previous Work



AoA representations do not have any dummy activities

The AoA networks are directed acyclic graphs with only one initial node and one and only one final node

The resource cost is fixed throughout the project execution

The resources are abundant

## Introduction – New Premises and Objectives

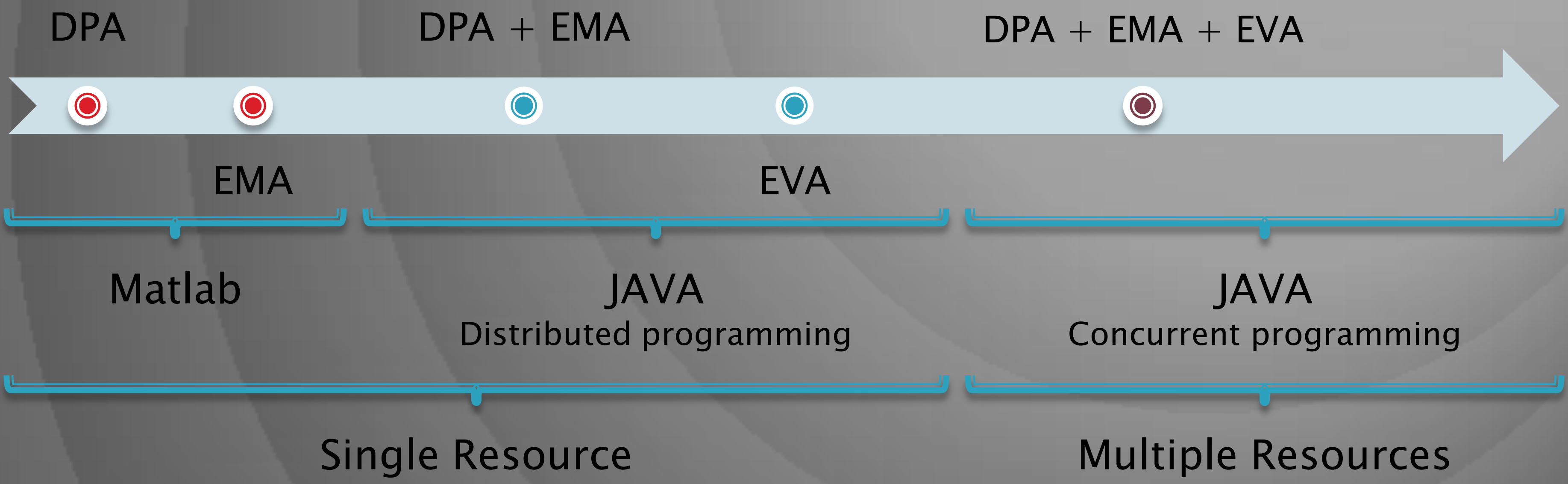
### New premises

- The project has a well determined set of resources available.
- The project activities can consume only a subset of the project resources.
- The resources are independent from each other.

### Objectives

- To develop new models involving multiple resources projects. One per each optimization method: dynamic programming, electromagnetic algorithm and evolutionary algorithm.
- Implementation of the new models in JAVA exploiting concurrent programming.

## Introduction – Implementation Timeline



**Legend**

DPA: Dynamic Programming Algorithm/Approach
EMA: Electromagnetic Algorithm
EVA: Evolutionary Algorithm

## Allocation Strategies – Behavior of an Activity

**SRPCO models**

(Single Resource Project Cost Optimization models)

- Each activity has its own work content

**MRPCO models**

(Multiple Resources Project Cost Optimization models)

- Each resource has its own work content

- In the MRPCO models, each resource will have its own work content and allocation constraints according to each activity's needs

## Allocation Strategies – Impact of Resource Multiplicity

The extension of the project cost evaluation from the SRPCO model is quite straightforward:

$$\mathrm{C} = \mathcal{E}\left[\sum_{a\in A}\sum_{r\in R_a}\left(c_r \times \mathrm{x}_r^a \times W_r^a\right) + c_L \times \max\left(0, \Upsilon_n - T\right)\right]$$

$\mathrm{A}$ : set of project activities

$c_r$ : quantity cost per unit of resource $r$

$\Upsilon_n$ : evaluated realization time of the last node

$W_r^a \sim \mathrm{Exp}\left(\lambda_r^a\right)$ : work content of resource $r$ on activity $a$

$\mathrm{R}_a$ : project resources subset needed by activity $a$

$\mathrm{x}_r^a$ : allocated quantity of resource $r$ on activity $a$

$T$ : schedule project realization time

## Allocation Strategies – Impact of Resource Multiplicity

To each resource allocation to an activity is associated an individual duration, evaluated similar to the SRPCO model

$$Y_r^a = \frac{W_r^a}{\mathrm{x}_r^a}$$

And the actual activity duration is the maximum of those individual ones

$$Y_a = \max_{r \in \mathrm{R}_a} \left( Y_r^a \right)$$

## Allocation Strategies – Impact of Resource Multiplicity

Clearly it makes little sense to expend more of a resource (and incur a higher cost) to have the activity duration under this resource less than its duration under another resource. Thus, it is desired to have

$$Y_i^a = Y_j^a, \forall i, j \in \mathrm{R}_a$$

Or, since there are random variables involved

$$\mathcal{E}\left[Y_i^a\right] = \mathcal{E}\left[Y_j^a\right], \forall i, j \in \mathrm{R}_a$$

To ensure allocation vectors leading to the desired equality, at least three strategies can be devised.

## Allocation Strategies – Quantity Oriented Strategy

• The QORAS (*Quantity Oriented Resource Allocation Strategy*) starts from the equality of individual durations in expectation.

• Rearranges the equation so that the resources are all expressed relative to one of them – the "base" resource.

$$\mathcal{E}\left[\frac{W_1^a}{x_1^a}\right] = \mathcal{E}\left[\frac{W_2^a}{x_2^a}\right] \Leftrightarrow x_2^a = \frac{\mathcal{E}\left[W_2^a\right]}{\mathcal{E}\left[W_1^a\right]} x_1^a;\ x_1^a \text{ base resource}$$

• The remaining allocations are immediately known through knowledge of the "base" resource allocation (considering expectations, there are no longer random variables).

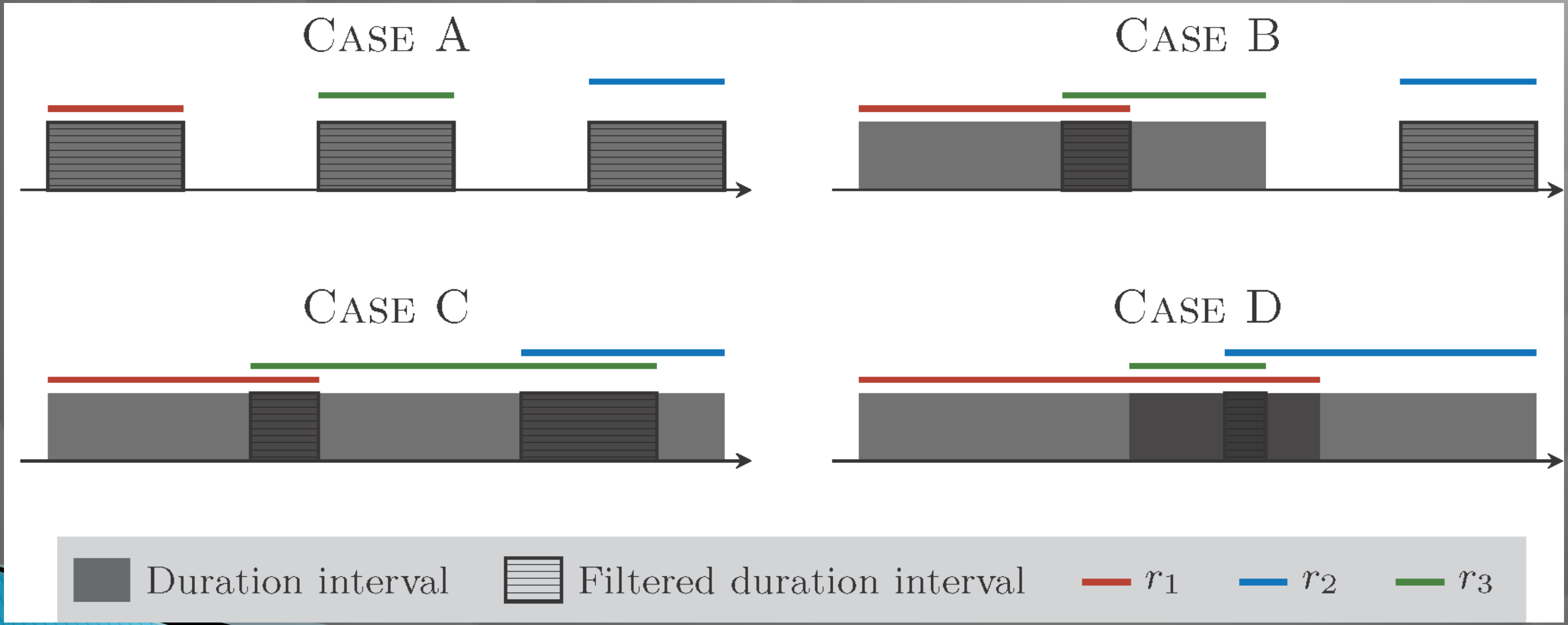
• Despite its simplicity and intuitive appeal, this strategy needs frequent corrections to the allocations in order to ensure that they remain in their feasible regions and the proportionality relations between them are preserved.

• This corrective mechanism becomes increasingly complex and hard to implement as the number of resources per activity increases. Furthermore, this strategy fails in situations where the desired equality is impossible to realize.

## Allocation Strategies – Duration Oriented Strategy

• The DORAS *(Duration Oriented Resource Allocation Strategy)* depends on sampling the work content. Then, the samples of the individual durations are determined by evaluating the possible durations according to the feasible allocation values.

• The possible common durations are evaluated via intersecting all the combinations of the sampled individual durations. Then, it filters those intervals leaving those that yielded more feasible resources.



## Allocation Strategies – Duration Oriented Strategy

• If the filtering results in just one interval the desired equality is fully satisfied. Else, it chooses the interval with the higher value.

• The durations of the selected interval are used to retrieve the allocation vector (of the involved resources) leading to each of them.

• Those resources not contributing to the selected interval, are put equal to their minimum values.

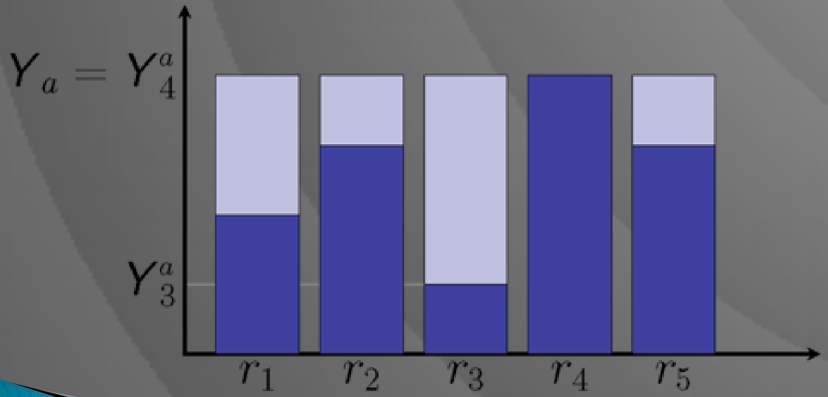
• This strategy is too complex to be implemented. Its algorithms experience exponential growth in both number of activities and resources. But, it copes rather well with any number of resources and with the cases when the equality of individual durations is impossible to achieve.

## Allocation Strategies – Waste Balanced Strategy

In the WBRAS (*Waste Balanced Resource Allocation Strategy*) we establish a mechanism that ensures always equal individual durations.

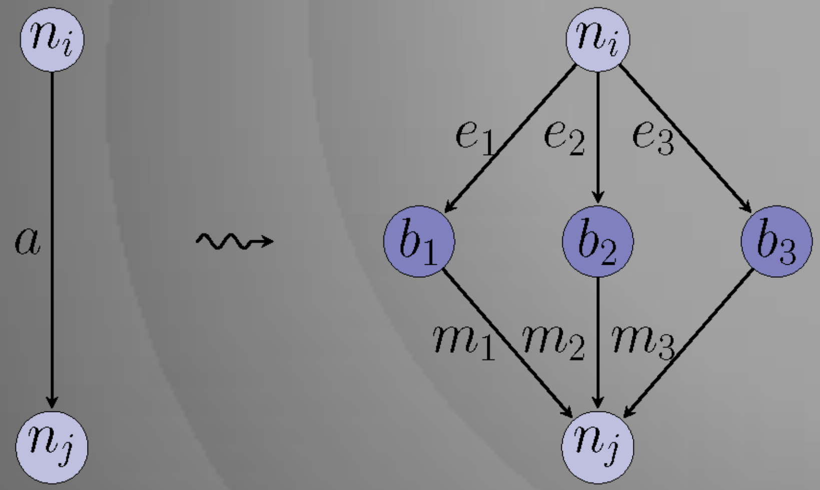
If we designate the time from the moment when a resource ceases to be needed to the end of the activity execution as “maintenance time”, then we are able to quantify the inequality of individual durations.

Each resource that lies unused may carry maintenance fees: storage, lifetime decline, etc.



| | |
|---|---|
| ▫ | maintenance phase |
| ■ | execution phase |
| $Y_3^a$ | duration yield by resource $r_3$ (execution) |
| $Y_4^a$ | duration yield by resource $r_4$ (execution) |
| $Y_a$ | duration of the activity |

## Allocation Strategies – Waste Balanced Strategy



We may conceptualize an activity as being a set of sub-activities in parallel, one per each resource, and composed of an execution phase followed by a maintenance phase.

The maintenance duration for a resource is simply the difference of the activity duration and the individual duration yield by it alone. In the figure we can see such an extended activity. Composed by 3 sub-activities (3 resources), each with the execution phase followed by the maintenance phase. The inner nodes mark the start of the idle time.

## Allocation Strategies – Waste Balanced Strategy

Waste Balanced Strategy

| Extended activity concept | New factor: maintenance |
|---|---|

Project Cost (sum of two components)

| Resource Allocation Cost | Resource Maintenance Cost |
|---|---|

New Assumption

This strategy must be applied to those projects being explicitly “maintenance aware”

## Optimization Models – Allocation Strategy Election

**QORAS**
- Intuitive
- Hard to generalize

**DORAS**
- Easy to generalize
- Exponential complexity

**WBRAS**
- Straightforward
- Introduces a new factor in the cost optimization

**Elected Strategy**

WBRAS – Waste Balanced Resource Allocation Strategy

## Optimization Models – Dynamic Programming Based

The maintenance cost of an activity closely depends from the activity's duration.

For an arbitrarily fixed activity allocation vector, we will describe this relationship through the following set

$$\Phi_a = \left\{ \left( \psi, \sum_{i \in \mathrm{R}_a} \left( s_i \left( \psi - \psi_i \right) \right) \right) \middle| \psi = \max_{i \in \mathrm{R}_a} \Big( \psi_i \Big) \ , \forall \psi_i \in \mathcal{Y}_i^a \right\}$$

$\mathcal{Y}_i^a$ : Sample of the individual duration yielded by resource $i$

$\psi$ : Sample value of the maximum of the individual durations – sample value of $Y_a$

$s_i$ : Maintenance cost per unit time of the resource $i$

## Optimization Models – Dynamic Programming Based

Suppose $x_1^a$, $x_2^a$, $x_3^a$ fixed and

$\mathcal{Y}_1^a = \{1,2\}$, $s_1 = 1$ $\quad$ $\mathcal{Y}_2^a = \{1,4\}$, $s_2 = 0.5$

$\mathcal{Y}_3^a = \{2,3\}$, $s_3 = 2$

| $\psi_1$ | $\psi_2$ | $\psi_3$ | $\psi = \max(\psi_1,\psi_2,\psi_3)$ | $\sum_{i=1}^{3} s_i(\psi-\psi_i)$ | $\Phi_a$ |
|---|---|---|---|---|---|
| 1 | 1 | 2 | 2 | $1(2-1)+0.5(2-1)+2(2-2)$ | $(2,1.5)$ |
| 1 | 1 | 3 | 3 | 3 | $(3,3)$ |
| 1 | 4 | 2 | 4 | 7 | $(4,7)$ |
| 1 | 4 | 3 | 4 | 5 | $(4,5)$ |
| 2 | 1 | 2 | 2 | 0.5 | $(2,0.5)$ |
| 2 | 1 | 3 | 3 | 2 | $(3,2)$ |
| 2 | 4 | 2 | 4 | 6 | $(4,6)$ |
| 2 | 4 | 3 | 4 | 4 | $(4,4)$ |

## Optimization Models – Dynamic Programming Based

• The set forms a sample of a distribution of the two parameters (activity duration and maintenance cost) and each sample value has the probability of realization equal to the probability of the first component (activity duration).

• The distribution approximates the real distribution and it improves with increase in the sample size.

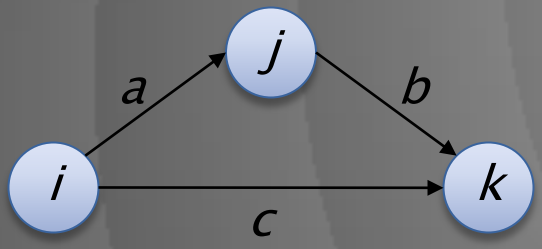
• Also, the distribution brings more complexity to the evaluation of the realization of a node, since we now have to deal not only with the realization time but with the corresponding maintenance cost.

• Thus, the states of the execution stages (dynamic programming) are vectors of pairs (realization time and maintenance cost).

Next, we present with an example the several repercussions of the above constructions.

## Optimization Models – Dynamic Programming Based

Node realization estimation



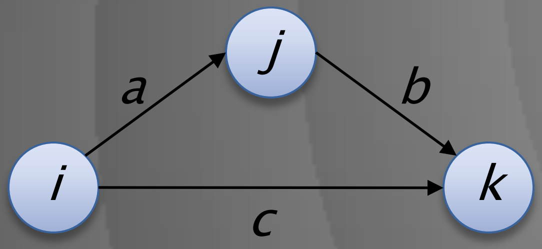
Let the picture aside be a portion of an AoA network and suppose that we are in a state composed by the nodes *i* and *j*. We want to estimate the realization (time and maintenance cost) of the node *k.*

To achieve that estimation we must know the distributions for the activities *b* and *c* which connect the state nodes to the target one. But then we need to specify that transition. Thus when, say, we go from node *j* to node *k* through activity *b* we must add the activity duration to the realization time of the node *j* and store the maintenance cost of the activity.

$$\Phi_{[n,a]} = \{(\alpha_n + \alpha_a, \beta_a) \,|\, \forall (\alpha_a, \beta_a) \in \Phi_a\}, \; (\alpha_n, \beta_n) \text{ state on node } n$$

## Optimization Models – Dynamic Programming Based

Node realization estimation



Suppose state $s_l = \{(\alpha_i, \beta_i), (\alpha_j, \beta_j)\}$:

$$(\alpha_i, \beta_i) = (7, 2)$$

$$(\alpha_j, \beta_j) = (4, 1)$$

$$\Phi_b = \{(1, 0), (2, 1)\}$$

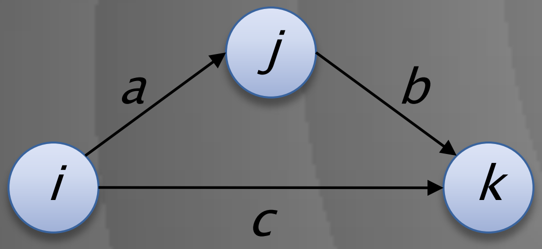
$$\Phi_c = \{(1, 1), (3, 0)\}$$

$$\Phi_{[i,c]} = \{(7 + 1, 1), (7 + 3, 0)\} = \{(8, 1), (10, 0)\}$$

$$\Phi_{[j,b]} = \{(4 + 1, 0), (4 + 2, 1)\} = \{(5, 0), (6, 1)\}$$

## Optimization Models – Dynamic Programming Based

### Node realization estimation



The actual estimation on the realization of the node will be determined from one of two scenarios:

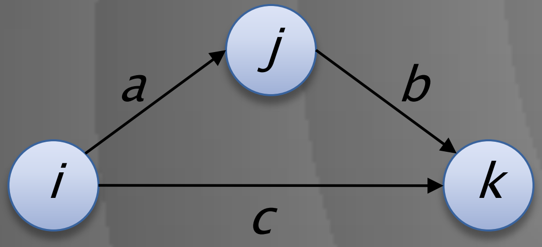
- The node is already contemplated on the state. Therefore nothing is to be determined (its realization is already known)

- The node is not contemplated on the state. In this case we must set its realization time as the maximum of all the "arriving" activities contributions and add all the maintenance costs from them.

$$\mathrm{DCmax}\left(\Phi_1, \ldots, \Phi_n\right) = \left\{\left(\max_{i=1}^{n}\left(\alpha_i\right), \sum_{i=1}^{n}\beta_i\right) \middle| \forall(\alpha_i, \beta_i) \in \Phi_i\right\}$$

## Optimization Models – Dynamic Programming Based

### Node realization estimation



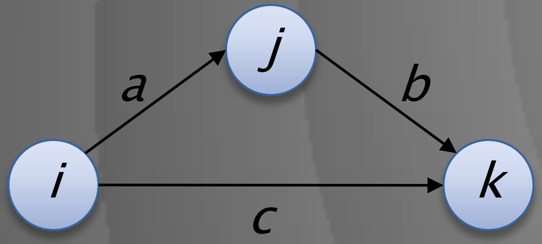
Let $P$ be the AoA network part preceding node $k$, with each element representing a predecessor node and the activity on the arc connecting it to node $k$.

In the example aside, that would be $P = \{(i,c),(j,b)\}$

$$r_k = \begin{cases} \{(\alpha_k, \beta_k)\} & \text{node } k \text{ contemplated on state } s_l \\ \underset{(n,a)\in P}{\mathrm{DCmax}} \left(\Phi_{[n,a]}\right) & \text{otherwise} \end{cases} \quad \text{, for a given state } s_l$$

## Optimization Models – Dynamic Programming Based

Node realization estimation



$$\Phi_{[i,c]} = \{(7+1,1),(7+3,0)\} = \{(8,1),(10,0)\}$$

$$\Phi_{[j,b]} = \{(4+1,0),(4+2,1)\} = \{(5,0),(6,1)\}$$

$$\Upsilon_k = \mathtt{DC}\max\left(\Phi_{[i,c]},\Phi_{[j,b]}\right)$$

$$= \left\{\left(\max\left(8,5\right),1+0\right),\left(\max\left(8,6\right),1+1\right),\ldots\right\}$$

$$= \{(8,1),(8,2),(10,0),(10,1)\}$$

## Optimization Models – Dynamic Programming Based

The quantity cost (the cost of resource utilization) of an activity is as follows

$$\mathcal{C}_{\mathrm{Q}}^{a} = \sum_{r \in \mathrm{R}_a} \left( c_r \times \mathrm{x}_r^a \times \mathcal{W}_r^a \right)$$

And for the special case of the “fixed activities” we have the following constant

$$\mathrm{rcf}_{\mathrm{Q}} = \mathcal{E}\left[ \sum_{a \in \mathcal{F}} \mathcal{C}_{\mathrm{Q}}^{a} \right] = \sum_{a \in \mathcal{F}} \sum_{r \in \mathrm{R}_a} \left( c_r \times \mathrm{x}_r^a \times \mathcal{E}\left[\mathcal{W}_r^a\right] \right), \ \mathcal{F} \text{ set of all “fixed activities”}$$

## Optimization Models – Dynamic Programming Based

The formula for the first stage is

$$f_1(s_1|\mathcal{F}) = \mathrm{rcf}_{\mathrm{Q}} + \min_{\mathrm{X}_1} \left( \mathcal{E}\left[ C_{\mathrm{Q}}^1 + \mathcal{E}\left[ \mathrm{sumpair}\left( c_L \dot{\times} U \right) \right] \right] \right)$$

Where

$$\mathrm{sumpair}\left\{(\beta_{11},\beta_{12}),\ldots,(\beta_{m1},\beta_{m2})\right\} = \left\{\beta_{11}+\beta_{12},\ldots,\beta_{m1}+\beta_{m2}\right\}$$

$$c_L \dot{\times} \left\{(\alpha_1,\beta_1),\ldots,(\alpha_m,\beta_m)\right\} = \left\{(c_L \times \alpha_1,\beta_1),\ldots,(c_L \times \alpha_m,\beta_m)\right\}$$

$$U = \mathtt{DC}\mathrm{max}\left(\overline{0}, \Upsilon_n - \overline{T}\right)$$

$$\overline{h} = \left\{(h,0)\right\},\ h \text{ constant and } (h,0) \text{ with probability } 1$$

## Optimization Models – Dynamic Programming Based

The formulas for the remaining stages are of the form

$$f_k(s_k|\mathcal{F}) = \sum_{(\alpha,\beta)\in s_k} \beta + \min_{\mathrm{X}_k} \left( \mathcal{E}\left[ C_{\mathrm{Q}}^{k} + \mathcal{E}\left[ f_{k-1}(s_{k-1}|\mathcal{F}) \right] \right] \right) \ , \ k > 1$$

And the main formula, assuming *K* stages and state *$s_K$=0* means zero duration and zero maintenance cost as it refers to the initial node

$$f(s_K = 0) = \min_{\mathcal{F}} \left( f_K(s_K|\mathcal{F}) \right)$$

## Optimization Models – Global Optimization Based

The GOA (*Global Optimization Algorithm*) implementations will rely on the *Monte Carlo Simulation* over the work contents. This will drop the random nature of the variables and the all process becomes quite straightforward.

The maintenance cost associated with an activity *a* is

$$C_{\mathrm{M}}^{a} = \sum_{r \in \mathrm{R}_a} \left(s_r \times \left(Y_a - Y_r^a\right)\right)$$

The GOA will minimize the following function

$$f = \sum_{a \in \mathrm{A}} \left(C_{\mathrm{Q}}^{a} + C_{\mathrm{M}}^{a}\right) + c_L \times \left(\max\left(0, \Upsilon_n - T\right)\right)$$

# JAVA Implementation

## Application

- The models were implemented in JAVA 1.6.
- A single command line application integrates all the models and execution options.

## AoA Representation

- Complete library for AoA network manipulation contemplating activities/projects with an arbitrary number of resources.

# JAVA Implementation

## Distributions

- Creation of approximate discrete distributions of a continuous one.
- Partition of a given interval.
- Sum and maximum of (discrete) distributions.

## Combinatorics

- Creation of classes providing linear addressing of all the possible combinations of a number of partitioned variables.
- Instead of computing complex nesting for cycles, we can simply go from the first combination to the last, deterministically and linearly.
- It is possible to process asynchronously subsets of the combinations: useful for distributed systems.

# JAVA Implementation

## Dynamic Programming Model Specifics

- The formulas were implemented in their natural recurrence order. That is the reverse order normally followed by the dynamic programming.

## Global Optimization Algorithm Model Specifics

- The matrix-like nature of the allocation vectors of all activities posed a problem. We unfolded those vectors on a single vector suitable for the algorithms. At the end, it is reconstructed back to the initial form.
- The EMA was translated from the initial imperative-oriented form to an object-oriented one. This allow easy addition of new features like the concurrent programming.

# JAVA Implementation

## Parallelism exploitation

- The DPA can be divided in several independent tasks: one per each allocation combination of the fixed activities. These tasks can be processed concurrently.
- The GOA involve several replication runs before giving the final (better) result. These replicas can be processed concurrently.
- With the GOA, the objective function value evaluation usually is not immediately required as other operations need to be performed first. Thus, we enable that evaluation to be asynchronous. Thus, the overall process does not need to stop while evaluating the objective value function (even if it is not immediately required).

## Results – Overall Configurations

• All tests were made with a computer with CPU Duo T7300 and 2GB RAM running Windows Vista. The JAVA virtual machine is version 1.6 at 64 bits.

• All tests still running after 5 hours were canceled.

• All the project resources have the allocation interval ranging from 0.5 to 1.5 units.

• The tests cover the DPA, EMA and EVA at both single-thread (linear programming) and multi-thread (concurrent programming).

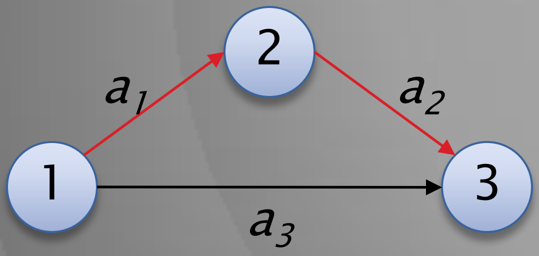
• For EMA and EVA two different configurations were used: one for sample size (k) of 500 and other with k=5000.

• The DPA used a sample size of 4 for the work contents.

• The results under single-thread and multi-thread were consistent with each other. For brevity, only one set of results is shown (those resulting from the single-thread runs).

• The allocation vectors components are in the same order as the index of the resources. On DPA the allocation vector of the decision activities are represented with braces, instead of parenthesis.

## Results – Project A.a (configuration)



| Resource | Lambda (per activity) | | |
|---|---|---|---|
| | $a_1$ | $a_2$ | $a_3$ |
| $r_1$ | 0.2 | 0.1 | 0.07 |

| Unit cost | |
|---|---|
| $c_r$ | $s_r$ |
| 1.0 | 0.0 |

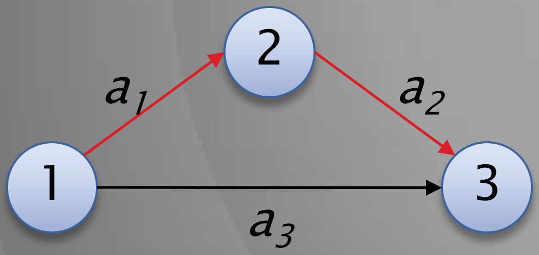
$T=16 \qquad c_L=2$

## Results – Project A.a (statistics and results)

| | DPA | EMA | | EVA | |
|---|---|---|---|---|---|
| | | k=500 | k=5000 | k=500 | k=5000 |
| Linear | 00:00:00.203 | 00:00:01.450 | 00:00:05.445 | 00:00:01.294 | 00:00:04.977 |
| Concurrent | 00:00:00.171 | 00:00:00.889 | 00:00:03.385 | 00:00:01.450 | 00:00:03.728 |

| | $a_1$ | $a_2$ | $a_3$ | Cost |
|---|---|---|---|---|
| DPA | {1.0} | | (1.0) | 43.7 |
| EMA 500 | (0.911) | (0.888) | (0.851) | 37.531 |
| EMA 5000 | (0.872) | (0.911) | (0.854) | 38.988 |
| EVA 500 | (0.874) | (0.887) | (0.84) | 37.291 |
| EVA 5000 | (0.892) | (0.879) | (0.841) | 38.68 |

## Results – Project A.b (configuration)



| Resource | Lambda (per activity) | | |
|---|---|---|---|
| | $a_1$ | $a_2$ | $a_3$ |
| $r_1$ | 0.07 | | 0.09 |
| $r_2$ | 0.1 | 0.04 | |
| $r_3$ | 0.2 | | |

| Unit cost | |
|---|---|
| $c_r$ | $s_r$ |
| 1.0 | 1.0 |
| 1.1 | 2.0 |
| 1.0 | 0.5 |

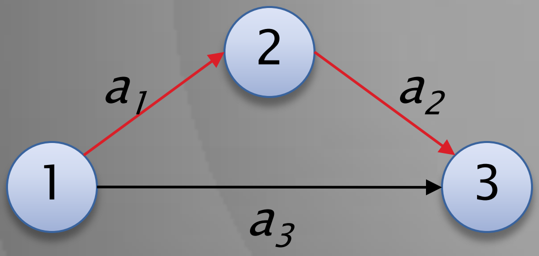
$T=41$ $c_L=2$

## Results – Project A.b (statistics and results)

| | DPA | EMA | | EVA | |
|---|---|---|---|---|---|
| | | k=500 | k=5000 | k=500 | k=5000 |
| Linear | 00:00:00.842 | 00:00:02.910 | 00:00:12.745 | 00:00:02.106 | 00:00:09.790 |
| Concurrent | 00:00:00.827 | 00:00:01.436 | 00:00:07.285 | 00:00:01.779 | 00:00:06.193 |

| | $a_1$ | $a_2$ | $a_3$ | Cost |
|---|---|---|---|---|
| DPA | {1.5,1.0,1.0} | | (0.5) | 113.147 |
| EMA 500 | (1.164,0.814,0.5) | (1.096) | (0.5) | 84.742 |
| EMA 5000 | (1.223,0.856,0.501) | (1.102) | (0.501) | 88.108 |
| EVA 500 | (1.195,0.836,0.5) | (1.087) | (0.5) | 84.335 |
| EVA 5000 | (1.118,0.782,0.5) | (1.113) | (0.5) | 87.234 |

## Results – Project A.c (configuration)



| Resource | Lambda (per activity) | | |
|---|---|---|---|
| | $a_1$ | $a_2$ | $a_3$ |
| $r_1$ | 0.1 | 0.03 | |
| $r_2$ | 0.2 | 0.06 | 0.07 |
| $r_3$ | | 0.03 | 0.09 |
| $r_4$ | 0.04 | 0.04 | 0.07 |

| Unit cost | |
|---|---|
| $c_r$ | $s_r$ |
| 1.0 | 1.0 |
| 1.1 | 2.0 |
| 1.0 | 0.5 |
| 2.0 | 0.1 |

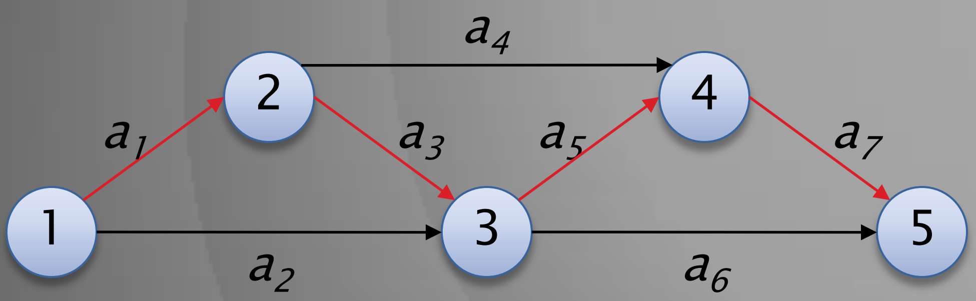
$T=61$ $c_L=2$

## Results – Project A.c (statistics and results)

| | DPA | EMA | | EVA | |
|---|---|---|---|---|---|
| | | k=500 | k=5000 | k=500 | k=5000 |
| Linear | > 5h | 00:00:07.660 | 00:00:58.812 | 00:00:03.978 | 00:00:29.160 |
| Concurrent | > 5h | 00:00:04.410 | 00:00:31.153 | 00:00:03.339 | 00:00:16.680 |

| | $a_1$ | $a_2$ | $a_3$ | Cost |
|---|---|---|---|---|
| DPA | | | | Aborted |
| EMA 500 | (0.568,0.5,1.421) | (1.072,0.547,1.072,0.804) | (0.64,0.504,0.64) | 274.8 |
| EMA 5000 | (0.557,0.5,1.393) | (1.053,0.536,1.054,0.79) | (0.621,0.507,0.622) | 281.107 |
| EVA 500 | (0.533,0.5,1.331) | (1.048,0.542,1.048,0.786) | (0.523,0.5,0.523) | 269.528 |
| EVA 5000 | (0.507,0.5,1.267) | (1.058,0.533,1.058,0.805) | (0.542,0.5,0.567) | 276.860 |

## Results – Project B.a (configuration)



| Resource | Lambda (per activity) | | | | | | |
|---|---|---|---|---|---|---|---|
| | $a_1$ | $a_2$ | $a_3$ | $a_4$ | $a_5$ | $a_6$ | $a_7$ |
| $r_1$ | 0.08 | 0.06 | 0.09 | 0.05 | 0.07 | 0.03 | 0.04 |

| Unit cost | |
|---|---|
| $c_r$ | $s_r$ |
| 1.0 | 0.0 |

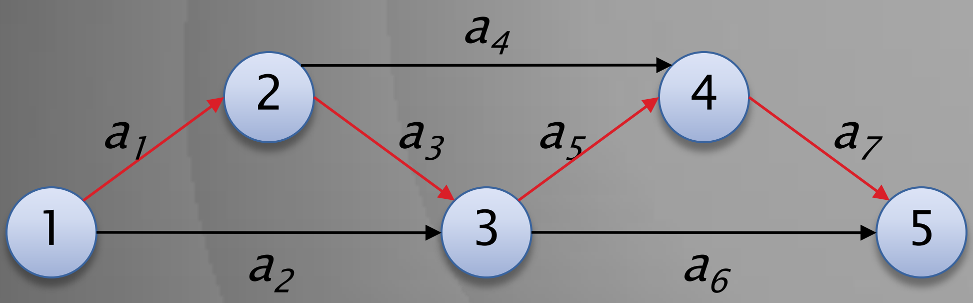
$T=66 \qquad c_L=5$

## Results – Project B.a (statistics and results)

| | DPA | EMA | | EVA | |
|---|---|---|---|---|---|
| | | k=500 | k=5000 | k=500 | k=5000 |
| Linear | 00:00:50.607 | 00:00:04.461 | 00:00:34.351 | 00:00:03.270 | 00:00:18.189 |
| Concurrent | 00:00:30.467 | 00:00:02.808 | 00:00:16.707 | 00:00:02.714 | 00:00:11.123 |

| | $a_1$ | $a_2$ | $a_3$ | $a_4$ | $a_5$ | $a_6$ | $a_7$ | Cost |
|---|---|---|---|---|---|---|---|---|
| DPA | {1.5} | (1.0) | | (1.0) | | (1.5) | | 205.541 |
| EMA 500 | (1.364) | (0.87) | (1.117) | (0.882) | (1.127) | (1.053) | (1.319) | 209.633 |
| EMA 5000 | (1.265) | (0.872) | (1.209) | (0.894) | (1.093) | (1.027) | (1.301) | 218.927 |
| EVA 500 | (1.415) | (0.939) | (1.249) | (0.891) | (1.064) | (1.024) | (1.316) | 209.182 |
| EVA 5000 | (1.432) | (0.925) | (1.201) | (0.886) | (1. 075) | (1.055) | (1.374) | 216.076 |

## Results – Project B.b (configuration)



| Resource | Lambda (per activity) | | | | | | |
|---|---|---|---|---|---|---|---|
| | $a_1$ | $a_2$ | $a_3$ | $a_4$ | $a_5$ | $a_6$ | $a_7$ |
| $r_1$ | 0.02 | 0.04 | | | 0.03 | 0.04 | 0.07 |
| $r_2$ | 0.08 | | 0.04 | 0.06 | 0.05 | | 0.08 |

| Unit cost | |
|---|---|
| $c_r$ | $s_r$ |
| 1.0 | 1.0 |
| 1.1 | 2.0 |

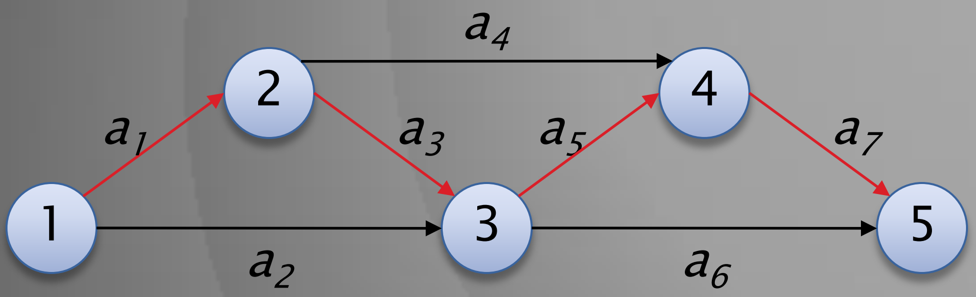
$T = 129 \quad c_L = 5$

## Results – Project B.b (statistics and results)

| | DPA | EMA | | EVA | |
|---|---|---|---|---|---|
| | | k=500 | k=5000 | k=500 | k=5000 |
| Linear | 00:42:07.293 | 00:00:07.846 | 00:01:05.208 | 00:00:04.524 | 00:00:35.318 |
| Concurrent | 00:25:52.130 | 00:00:04.337 | 00:00:33.150 | 00:00:03.572 | 00:00:19.516 |

| | $a_1$ | $a_2$ | $a_3$ | $a_4$ | $a_5$ | $a_6$ | $a_7$ | Cost |
|---|---|---|---|---|---|---|---|---|
| DPA | {1.5,0.75} | (1.0) | | (0.5) | | (1.0) | | 393.297 |
| EMA 500 | (1.5,0.5) | (0.5) | (1.5) | (0.5) | (1.339,0.804) | (0.658) | (1.104,0.966) | 361.883 |
| EMA 5000 | (1.5,0.5) | (0.5) | (1.5) | (0.5) | (1.272,0.763) | (0.643) | (1.146,1.002) | 376.068 |
| EVA 500 | (1.5,0.5) | (0.5) | (1.499) | (0.5) | (1.274,0.764) | (0.659) | (1.22,1.068) | 360.236 |
| EVA 5000 | (1.5,0.5) | (0.5) | (1.484) | (0.5) | (1.282,0.769) | (0.651) | (1.191,1.043) | 374.425 |

## Results – Project B.c (configuration)



| Resource | Lambda (per activity) | | | | | | |
|---|---|---|---|---|---|---|---|
| | $a_1$ | $a_2$ | $a_3$ | $a_4$ | $a_5$ | $a_6$ | $a_7$ |
| $r_1$ | | 0.04 | | | 0.02 | 0.04 | 0.07 |
| $r_2$ | 0.02 | | | 0.07 | | 0.04 | |
| $r_3$ | | | 0.03 | 0.09 | 0.05 | 0.024 | |

| Unit cost | |
|---|---|
| $c_r$ | $s_r$ |
| 1.0 | 1.0 |
| 1.1 | 2.0 |
| 1.0 | 0.5 |

$T=155$ $c_L=5$

## Results – Project B.c (statistics and results)

| | DPA | EMA | | EVA | |
|---|---|---|---|---|---|
| | | k=500 | k=5000 | k=500 | k=5000 |
| Linear | > 5h | 00:00:10.904 | 00:01:32.493 | 00:00:05.522 | 00:00:48.126 |
| Concurrent | > 5h | 00:00:05.709 | 00:00:49.470 | 00:00:03.915 | 00:00:24.757 |

| | $a_1$ | $a_2$ | $a_3$ | $a_4$ | $a_5$ | $a_6$ | $a_7$ | Cost |
|---|---|---|---|---|---|---|---|---|
| DPA | | | | | | | | Aborted |
| EMA 500 | (1.5) | (0.5) | (1.5) | (0.667,0.519) | (1.436,0.575) | (0.564,0.564,0.939) | (1.429) | 429.936 |
| EMA 5000 | (1.5) | (0.5) | (1.499) | (0.656, 0.51) | (1.453,0.582) | (0.575,0.575,0.958) | (1.45) | 447.548 |
| EVA 500 | (1.494) | (0.5) | (1.499) | (0.679,0.529) | (1.452,0.681) | (0.575,0.575,0.958) | (1.421) | 433.6 |
| EVA 5000 | (1.5) | (0.5) | (1.5) | (0.653,0.51) | (1.427,0.571) | (0.548,0.548,0.914) | (1.359) | 441.731 |

## Conclusions

• For small networks with few resources, the DPA is acceptable and even faster than the GOA.

• The DPA quickly rises to run times above 5 hours, while the GOA stay within few seconds to a couple of minutes (even for large *k*).

• The EVA is the algorithm with better overall performance.

• All the implementations (DPA and GOA) achieve better performance when they are executed on a concurrent platform.

• The resulting allocations and objective function values obtained by the several algorithms are consistent with each other.

• The large *k=5000* showed no improvement on the results of GOA.

• The actual implementation of DPA is not suitable for practical tests; while the GOA represent a good alternative in both performance and results.

## References

[1] R. Moutinho. Gestão de Projectos – Alocação de Múltiplos Recursos. Relatório de estágio, Universidade do Minho, December 2007.

[2] A. P. Tereso, M. M. T. Araújo, and S. E. Elmaghraby. Adaptive Resource Allocation in Multimodal Activity Networks. *International Journal of Production Economics*, 92(1):1–10, November 2004.

[3] A. P. Tereso, M. M. T. Araújo, and S. E. Elmaghraby. The Optimal Resource Allocation in Stochastic Activity Networks via the Electromagnetism Approach. *Ninth International Workshop on Project Management and Scheduling (PMS'04)*, April 2004.

[4] A. P. Tereso, J. R. M. Mota, and R. J. T. Lameiro. Adaptive Resource Allocation to Stochastic Multimodal Projects: A Distributed Platform Implementation in Java. *Control and Cybernetics Journal*, 35(3):661–686, 2006.

[5] A. P. Tereso, R. A. Novais, and M. M. T. Araújo. The Optimal Resource Allocation in Stochastic Activity Networks via the Electromagnetism Approach: A Platform Implementation in Java. Reykjavíc, Iceland, July 2006. 21st European Conference on Operational Research (EURO XXI). Submitted to the "Control and Cybernetics Journal" (under revision).

[6] A. P. Tereso, L. A. Costa, R. A. Novais, and M. T. Araújo. The Optimal Resource Allocation in Stochastic Activity Networks via the Evolutionary Approach: A Platform Implementation in Java. Beijing, China, May 30 – June 2 2007. International Conference on Industrial Engineering and Systems Management (IESM' 2007). Full paper published in the proceedings (ISBN 978-7-89486-439-0) and submitted to the "Computers and Industrial Engineering".