Universidade do Minho
Escola de Engenharia
Departamento de Produção e Sistemas
Portugal

IESM 2009 · May 13-15
Montréal · CANADA

A-o-A Editor

Tool to edit and visualize Activity-on-Arc networks

Anabela Pereira Tereso (anabelat@dps.uminho.pt)
João Paulo Cardoso (cardoso.paulo@gmail.com)
José Pedro Cabral (jpedrocabral@gmail.com)

Topics

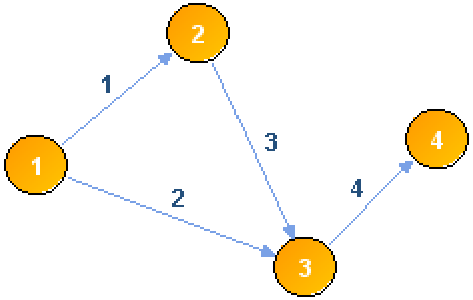
- Introduction
- Activity-on-Arc Networks
- Application Data
- Libraries
- Application design
- Demonstration
- References

Introduction

- Integrated, along with several other projects, in a study that does research in the field of Project Management
- Necessity to allow the visualization of networks created in the projects of the refered study
- Two types of A-o-A networs to work with:
  - Single Resource
  - Multiple Resources

Acticity-on-Arc Networks

- Specifies the tasks that have to be concluded in order to complete the project



- Nodes mark the beginning and ending of each acticity
- Precedence of activities must be taken into account

Aplication Data

- Two types of networks stored in two types of files
- TXT file (single resource network):
  - Line 1 – number of arcs and nodes
  - Line 2 to n-2 – one activity per line (starting node, destination node, lambda, min and max)
  - Line n-1 – delay cost per unit
  - Line n – project due time

Aplication Data

- XML File (multiple resources networks)
  - Project name
  - Project’s due time and delay cost per unit
  - Resources (id, min, max, qcu, mcu)
  - Activities (id, starting and destination nodes)
    - Resources used by the activity (id, lambda, and occasionally changes to the default resource data)

Libraries

- Two libraries were used:

- JGraph
  - ▫ Open Source Java library
  - ▫ Graph visualization
  - ▫ Compatible with Java Swing
  - ▫ Provides many solutions and is quite extensive
  - ▫ Allows visual customization

Libraries

- JGrapht
  - Open Source Java library
  - Facilitates the manipulation of graph data
  - Easily integrated with JGraph
  - Enables the visualization of graphs

Application Design

- How to store application data?
- Three HashMaps created to store Nodes, Activities and Resources
- Another HashMap to store resources associated with each activity

Application Design

- Data related to the position and the dimension of nodes had to be stored in a separate file
- Extra coordinates had to be stored to preserve the curves in the arcs
- The applications developed previously weren’t prepared to deal with these constraints

Application Design

- AoA Files
  - Based in XML format
  - Stores nodes's coordinates and dimensions
  - Stores extra coordinates for each arc's curvature

- Export graphical layout to image formats suitable for Word and Latex use
  - PNG and EPS

Application Design

- Runs in two languages (English and Portuguese)
- The first menu allows us to:
  - Start a new single resource or multiple resource project
  - Open the two mentioned projects
  - Open a project with a graphical layout

Application Design

- The main interface allows:
  - Insertion and removal of nodes and activities
  - Saving the Network in two formats (TXT and XML)
  - Export to EPS and PNG
  - Changing the zoom
  - Changing the layout of the nodes
  - Modifying the arcs curvatures

Application Design

- Two additional interfaces
- One edits project resources
- The other edits resources associated with the activities

Application Design

- Three parsers were designed to read the data from the three file formats (TXT, XML an AoA)
- Another three parsers were design to write the data into the three file formats

Application Design

- Some important classes of the JGraph library:
  - **JGraph** – main class where all the graph information is stored (nodes, arcs and graphical atributes)
  - **GraphSelectionListener** – checks selected objects
  - **GraphUndoManager** – stores the state of the graph when there are changes in the layout

Application Design

- Some important classes of the JGrapht library:
  - **DefaultListenableGraph** – Type of graph that allows the use of listeners
  - **DefaultDirectedGraph** – Type of graph that allows external data (ex. from file) to be inserted **JGraphModelAdapter** – This class allows the two previous classes to be converted and used by JGraph (in this aplication only the DefaultListenable graph is used)

Application Design

- Method testProject()
  - Validates the project (checks for cycles and also if there is only one starting and ending nodes)
  - Calculates the projects expected duration (Critical Path Method)
- Class CoordCalc
  - Used to allocate nodes in the layout
  - This allocation only takes into account the minimizations of the intersections between nodes

Demonstration

- Creating Single Resource Network
- Opening created files (.AoA, .txt)
- Creating Multiple Resources Network
- Opening created files (.AoA, .xml)
- Editing Multiple networks

References

- Benson, D. (2007), “JGraph and JGraph Layout Pro User Manual”, (http://www.jgraph.com/support.html)
- Demeulemeester, E., Dodin, B., Herroelen, W. (1993) “A Random Activity Network Generator”, Operations Research, Vol. 41, nº 5, pp 972-980.
- Tereso, A.P., Araújo, M.M., Elmaghraby, S.E. (2004) “Adaptive Resource Allocation in Multimodal Activity Networks”, International Journal of Production Economics, Elsevier, Vol 92, Issue 1, pp. 1-10, November 2004.
- Tereso, A.P., Araújo, M.M., Elmaghraby, S.E. (2004a) “The Optimal Resource Allocation in Stochastic Activity Networks via the Electromagnetism Approach”, Ninth International Workshop on Project Management and Scheduling (PMS’04), Nancy, France, April 26-28, 2004.
- Tereso, A. P., Mota, J. R. M., Lameiro, R. J. T. (2006) “Adaptive Resource Allocation to Stochastic Multimodal Projects: A distributed platform implementation in Java”, Control and Cybernetics Journal, Vol. 35, Issue 3, pp. 661-686, 2006.
- Tereso, A.P., Araújo, M.M., Moutinho, R., Elmaghraby, S.E. (2008) “Project Management: multiple resources allocation”, to be presented in the International Conference on Engineering Optimization (EngOpt2008), Rio de Janeiro - Brazil, June 1-5, 2008.