A Lookahead Heuristic for the Minimization of Open Stacks Problem

Marco A. M. Carvalho
mamc@iceb.ufop.br
Federal University of Ouro Preto - Brazil

Nei Y. Soma
soma@ita.br
Technological Institute of Aeronautics - Brazil

OR54 Annual Conference – Edinburgh, UK
04-06 September 2012

Problem Description

Motivation

Example

INTRODUCTION

Introduction

- A factory manufactures different types of products in batches;
- Customers place orders for different products
  - The contents of each order are placed in a separated stack during manufacturing;
  - When the stack receives the first product, it is *opened*;
  - When the stack receives the last product , it is *closed*
    - The products are delivered;
    - The space is freed.

Introduction

- There is a limitation on the physical space used in the production environment
  - There is not enough space for all customer's stacks to be opened simutaneously;
  - If the number of open stacks increases beyond the available space, stacks must be removed in order to give space to the new stacks.
- The sequence in which the products are manufactured can reduce the maximum number of simultaneously open stacks
  - This is the aim of the *Minimization of Open Stacks Problem* (MOSP).

Motivation

- The problem is NP-Hard and has a variety of equivalent problems:
  - Cutting stock
    - Cutting Patterns Sequencing.
  - VLSI design
    - Gate Matrix Layout Problem;
    - PLA Folding.
  - Graph Problems
    - Pathwidth;
    - Interval Thickness;
    - Node Search Game;
    - Narrowness;
    - Split bandwidth;
    - Edge and Vertex Separation.

Example #1

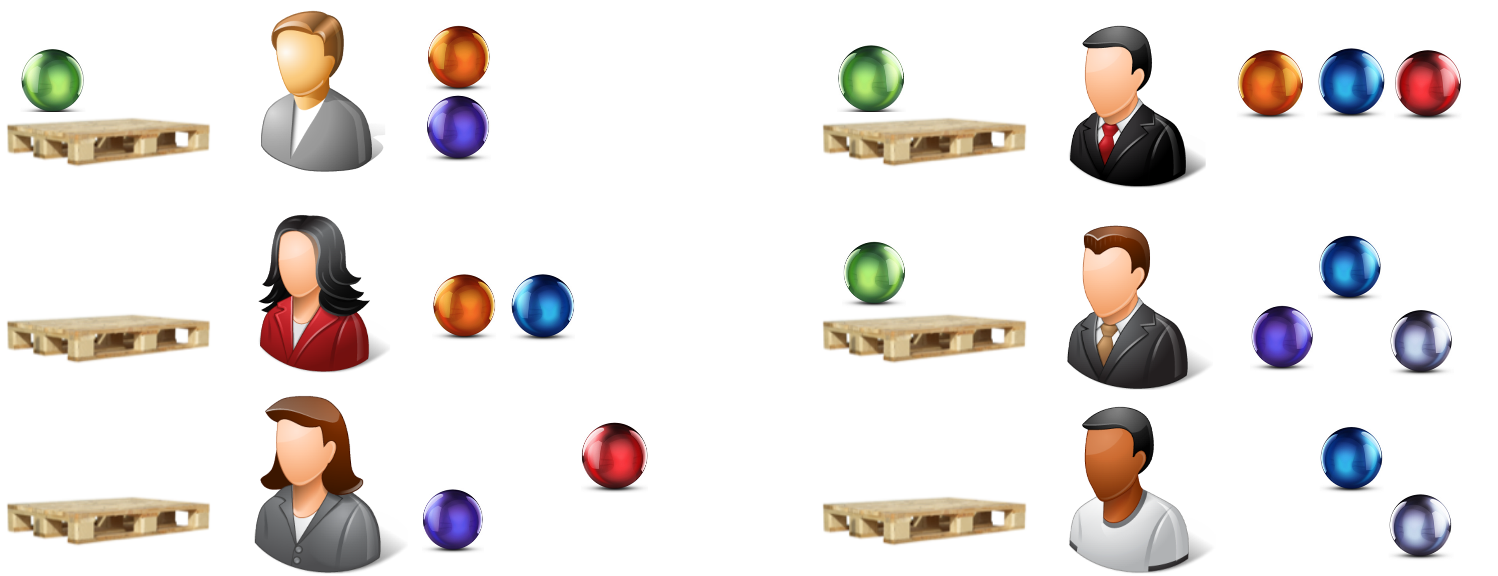
- Six customers;
- Six product types.

Example #1

- Manufacturing Sequence:
- Open Stacks: 0

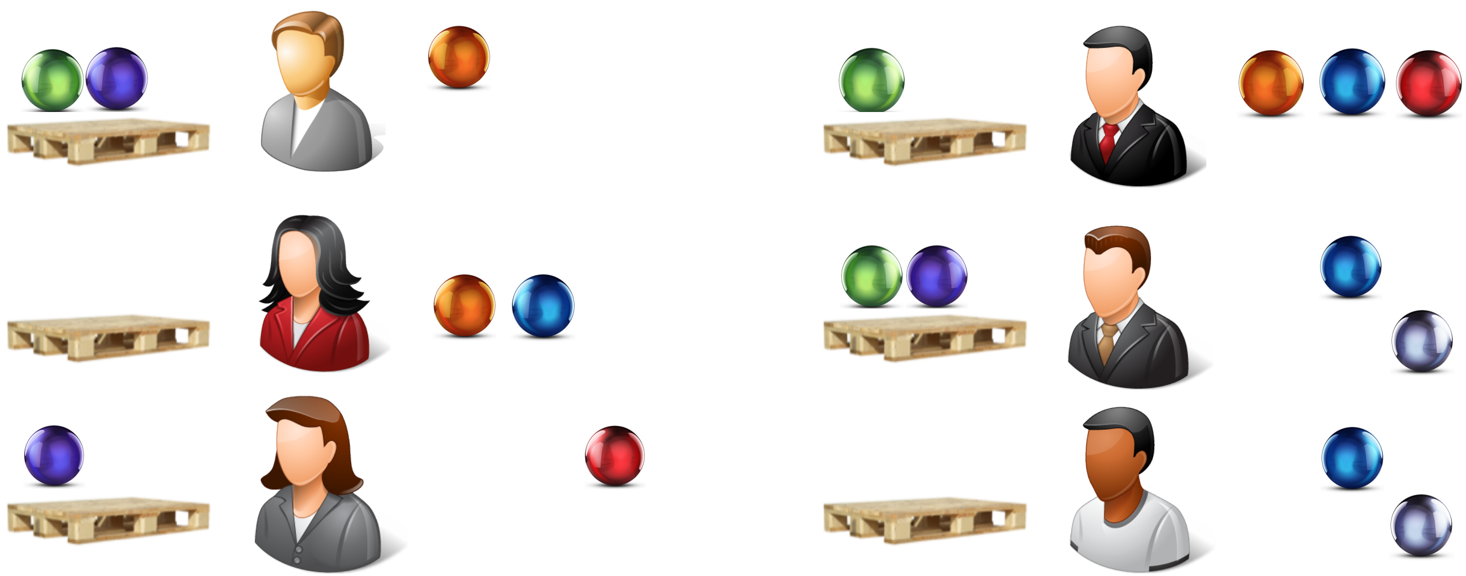
Example #1

- Manufacturing Sequence:
- Open Stacks: 3



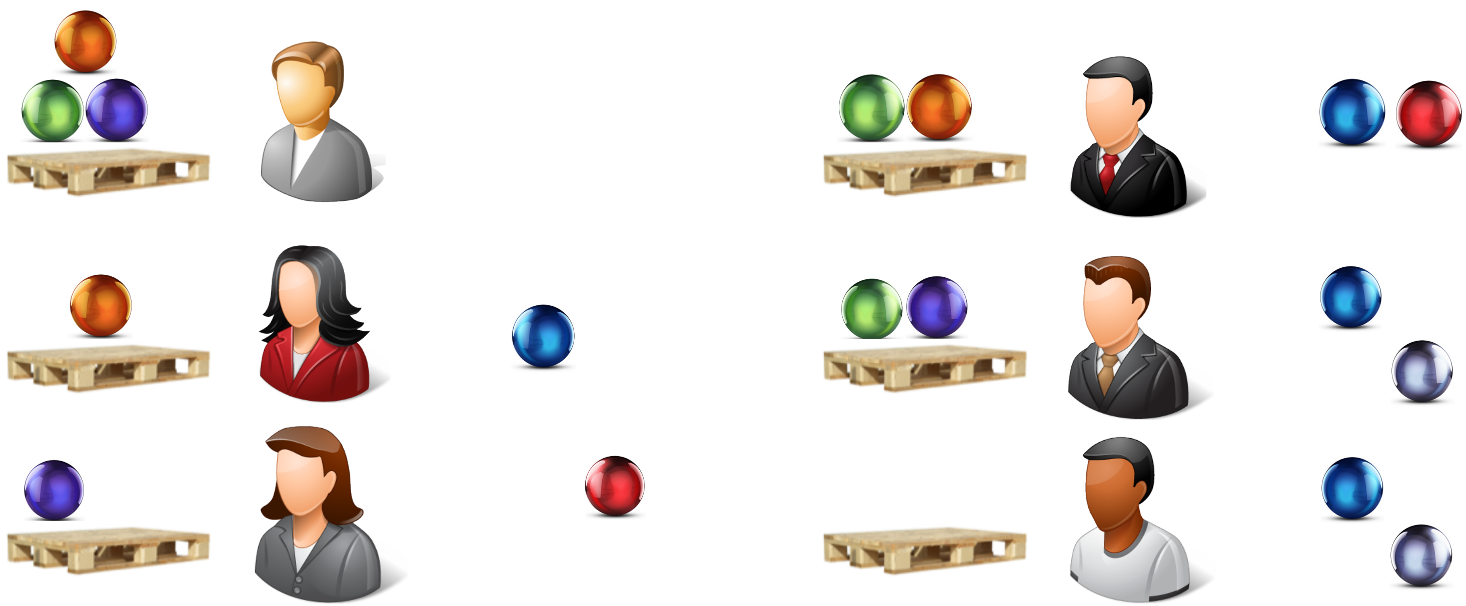
Example #1

- Manufacturing Sequence: 
- Open Stacks: 4

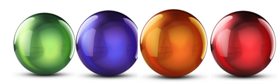


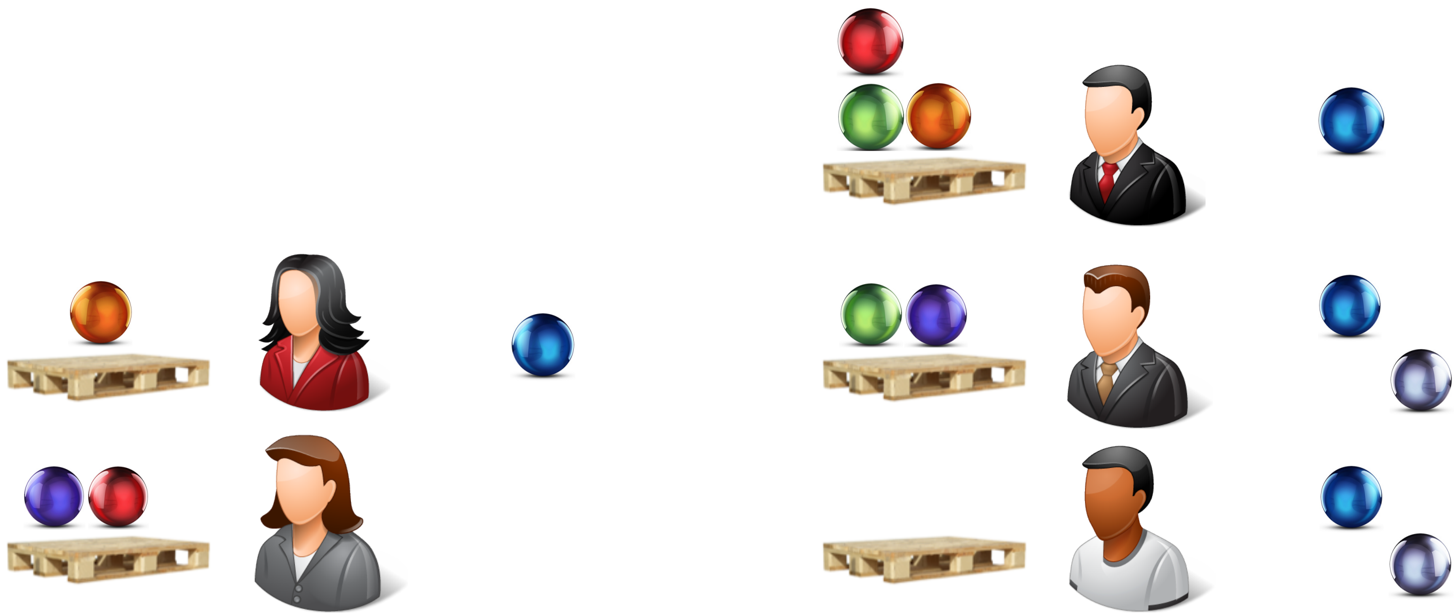
Example #1

- Manufacturing Sequence: 
- Open Stacks: **5**

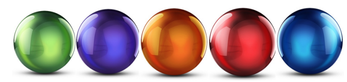


Example #1

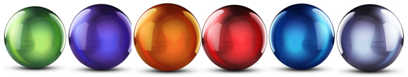
- Manufacturing Sequence: 
- Open Stacks: 4

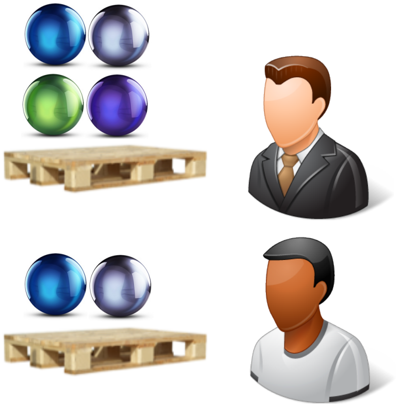


Example #1

- Manufacturing Sequence: 
- Open Stacks: 4

Example #1

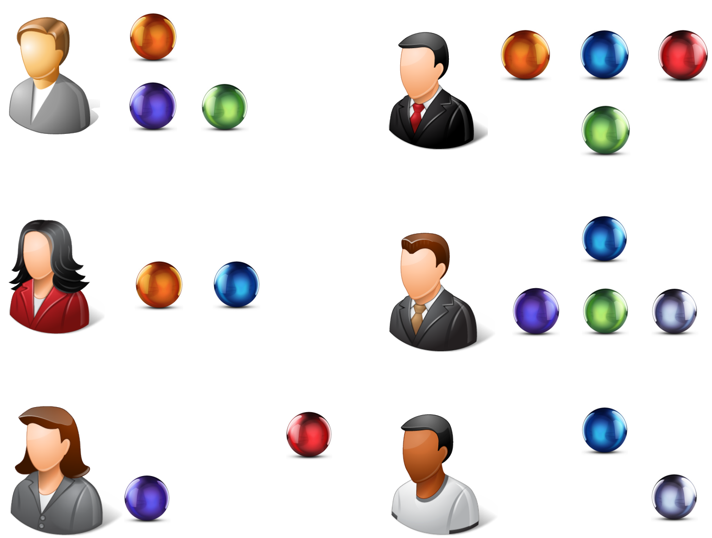
- Manufacturing Sequence: 
- Open Stacks: 2



Introduction

- Formally, given a boolean matrix ***M***:
  - Rows correspond to customer's orders;
  - Columns correspond to products;
  - $m_{ij} = 1$ iff order i contains product j;
  - $m_{ij} = 0$ otherwise;
  - Stacks are associated to rows
    - First product is manufactured: stack *opened*;
    - Last product is manufactured: stack *closed*;
- The objective is to find a permutation of columns such that the maximum number of open stacks is minimized.

Example #1 Revisited



	p1	p2	p3	p4	p5	p6
c1	1	0	0	1	1	0
c2	1	1	0	0	0	0
c3	0	0	1	1	0	0
c4	1	1	1	0	1	0
c5	0	1	0	1	1	1
c6	0	1	0	0	0	1

Example #1 Revisited

Manufacturing Sequence

Stacks

	p2	p4	p5	p1	p3	p6
c1		1	1	1		
c2	1	--	--	1		
c3		1	--	--	1	
c4	1	--	1	1	1	
c5	1	1	1	--	--	1
c6	1	--	--	--	--	1

Max Open Stacks: 6

Manufacturing Sequence

Stacks

	p6	p2	p1	p3	p4	p5
c1			1	--	1	1
c2		1	1			
c3				1	1	
c4		1	1	1	--	1
c5	1	1	--	--	1	1
c6	1	1				

Max Open Stacks: 4

THE OR SOCIETY

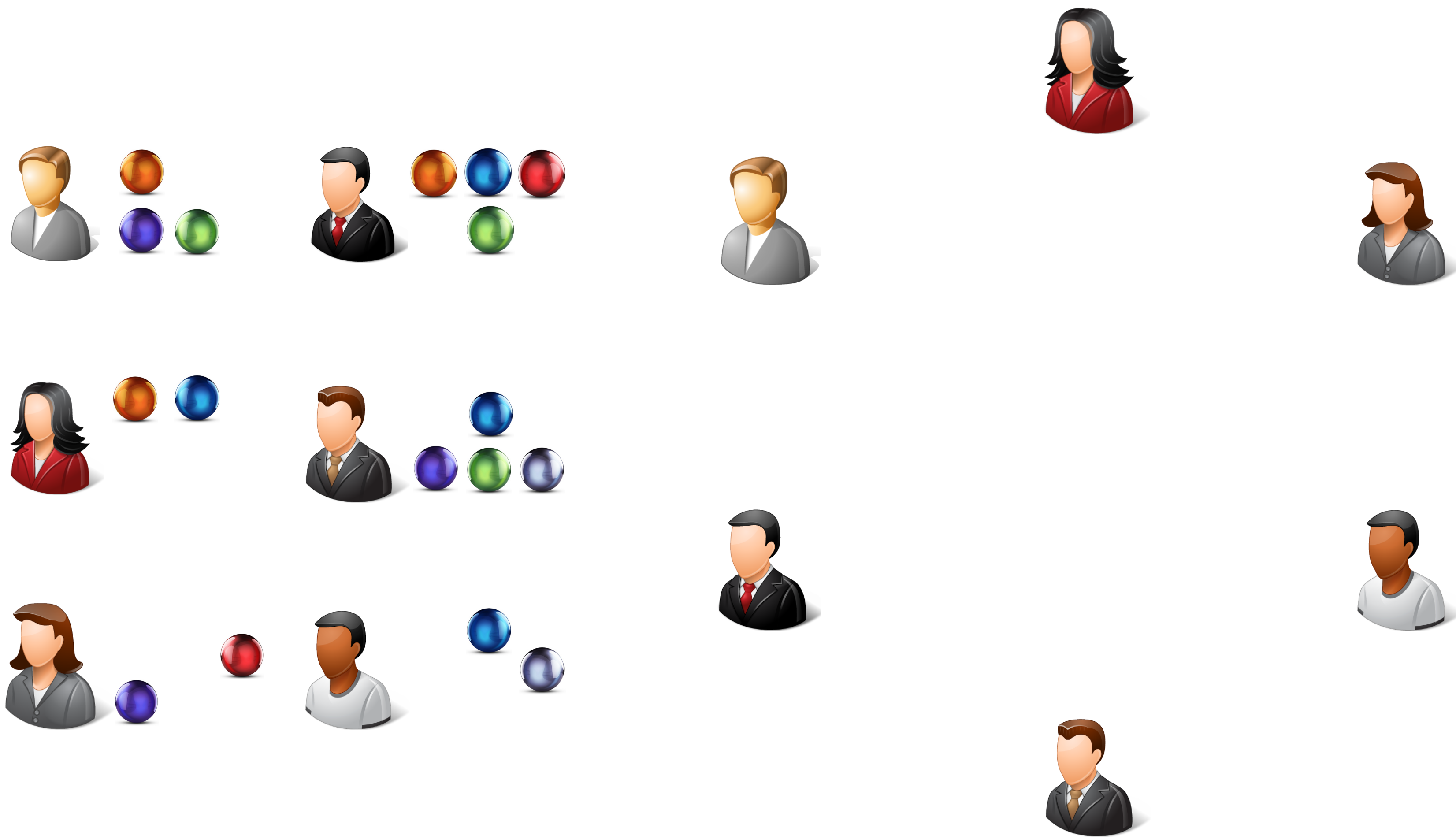
Representation
Preprocessing
Breadth-First Search
Products Sequencing
Improvement Rules

A LOOKAHEAD HEURISTIC

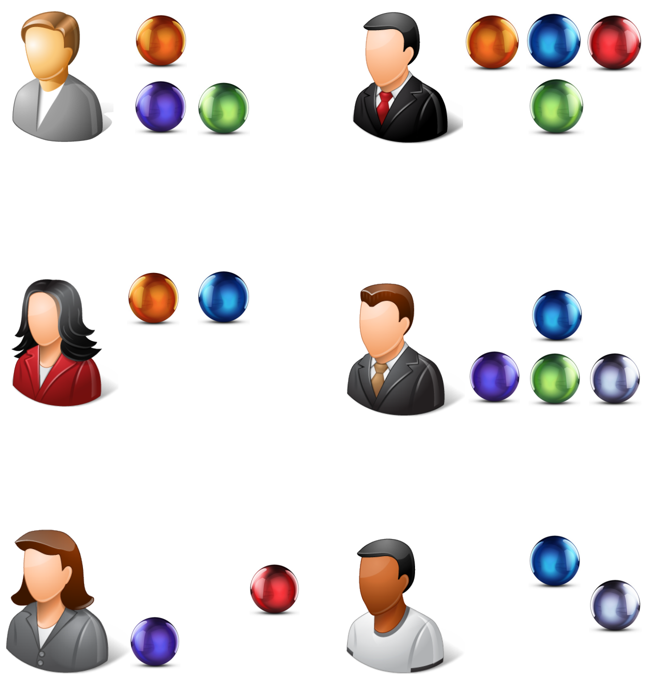
Representation

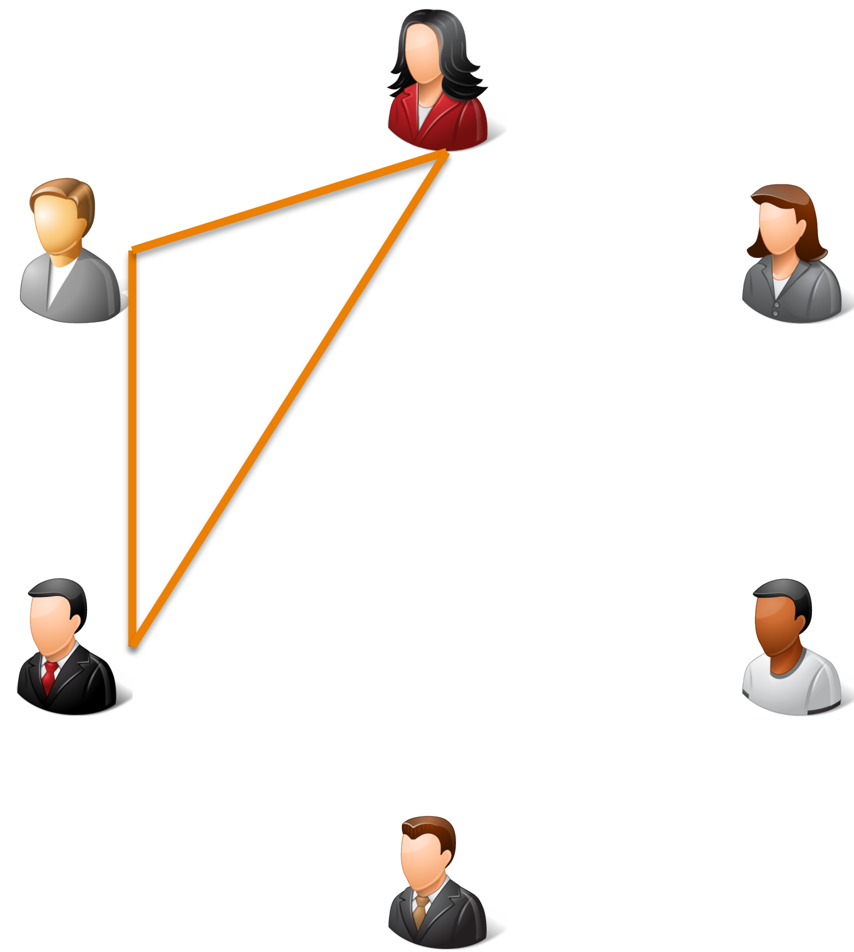
- In MOSP graphs, nodes correspond to customer's orders
  - Edges connect customers that ordered at least one product in common;
  - Multiple edges and loops are not considered;
  - Each product produces a clique in the graph;
  - There are polynomial algorithms for some special topologies.

MOSP Graph



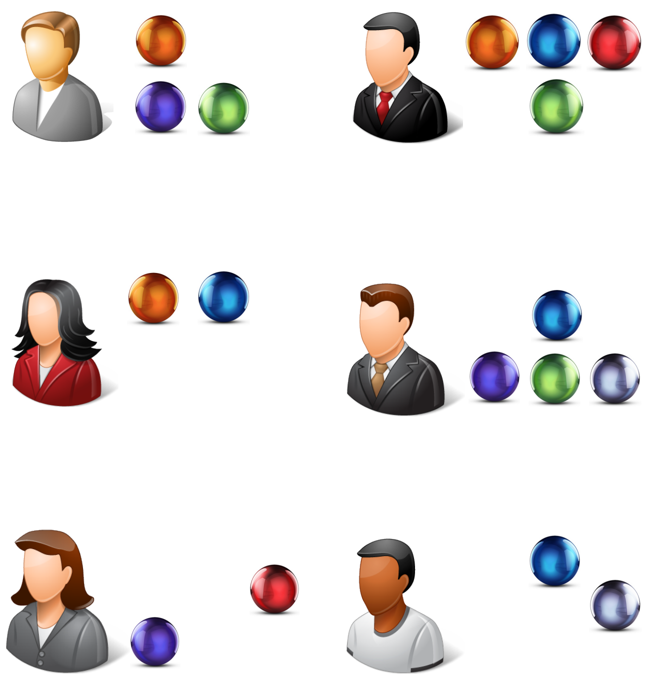
MOSP Graph

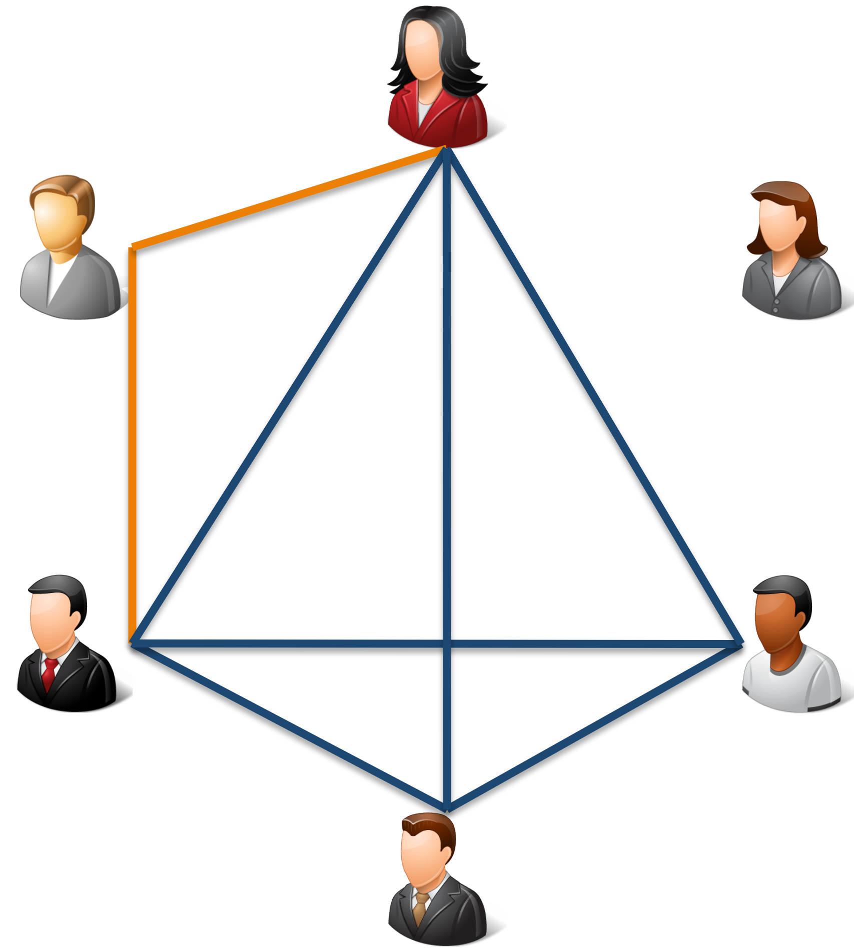




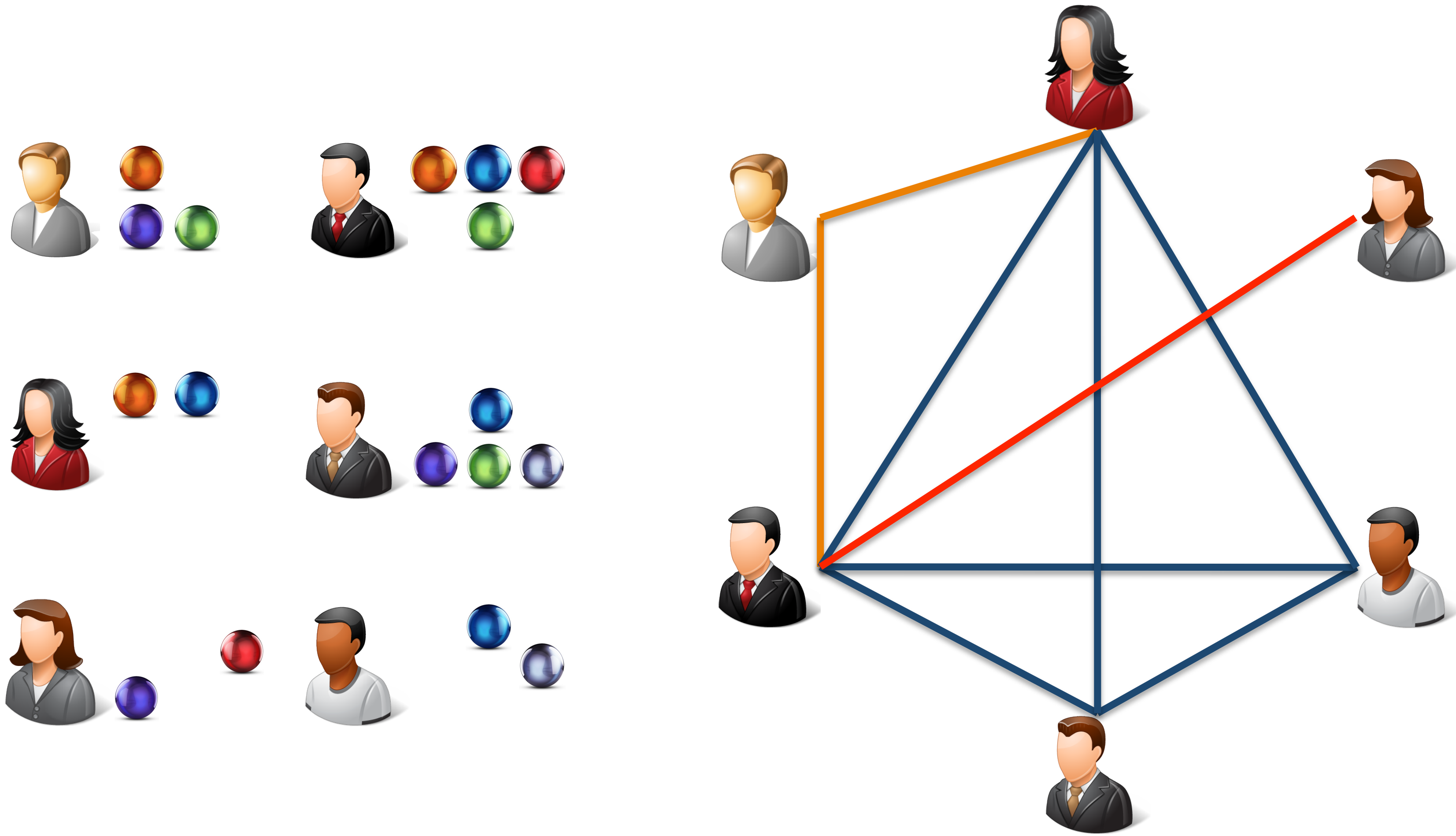
THE OR SOCIETY

MOSP Graph

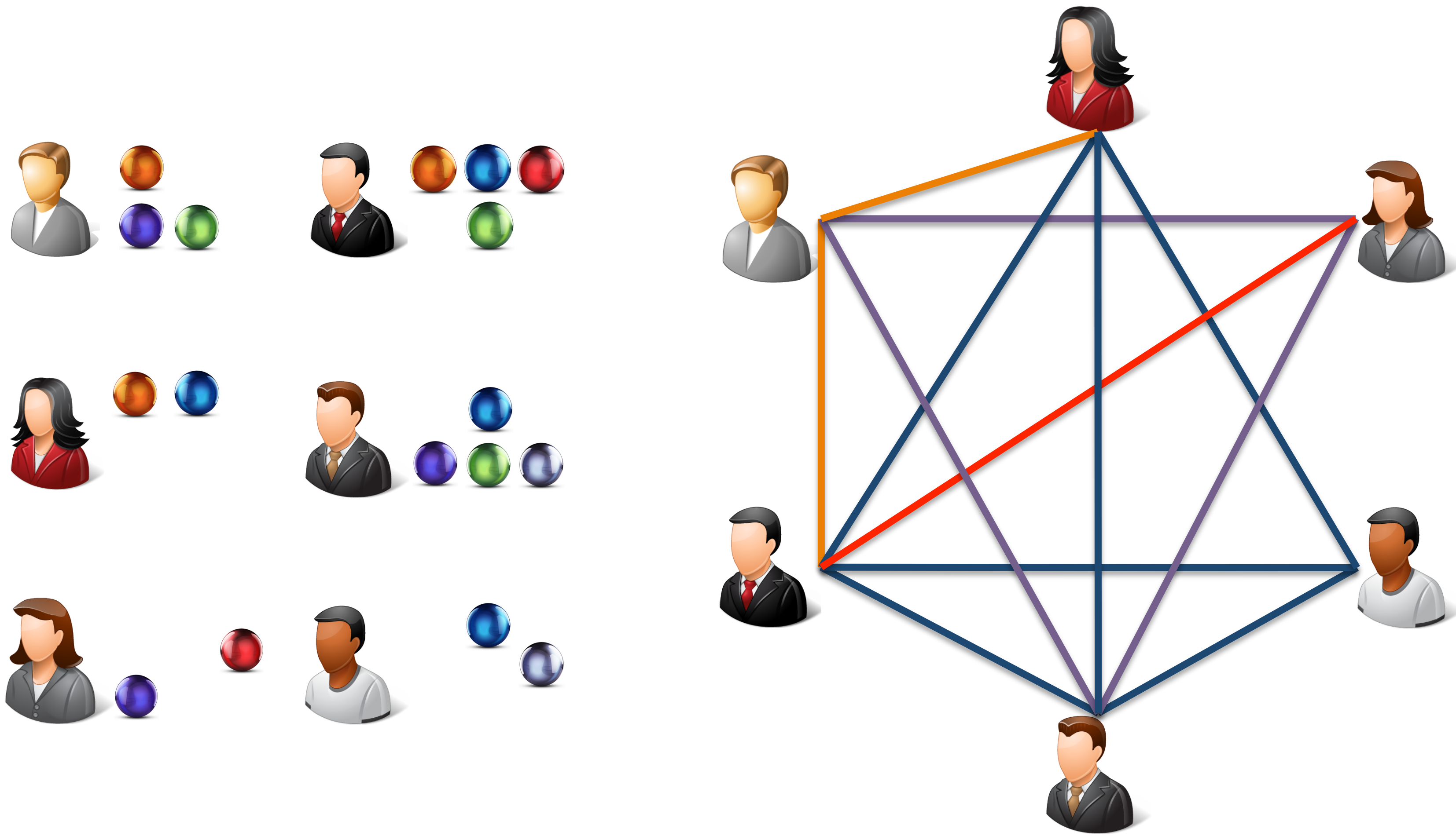




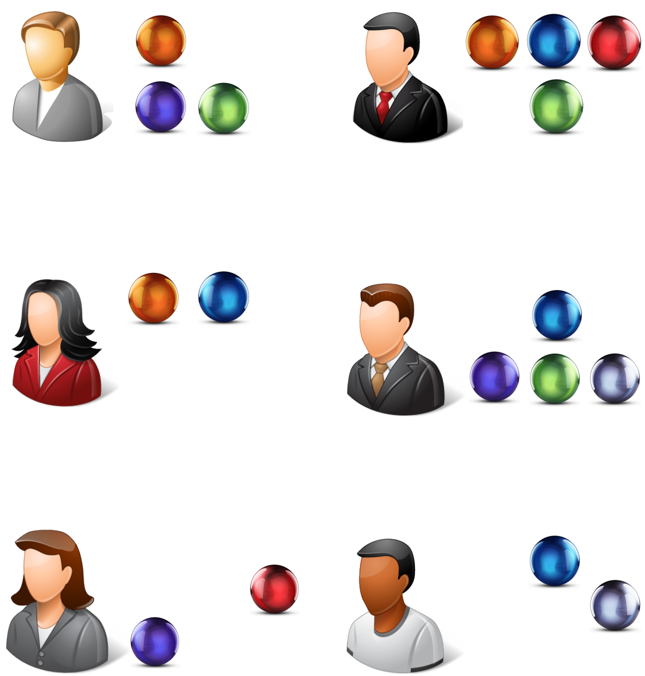
MOSP Graph

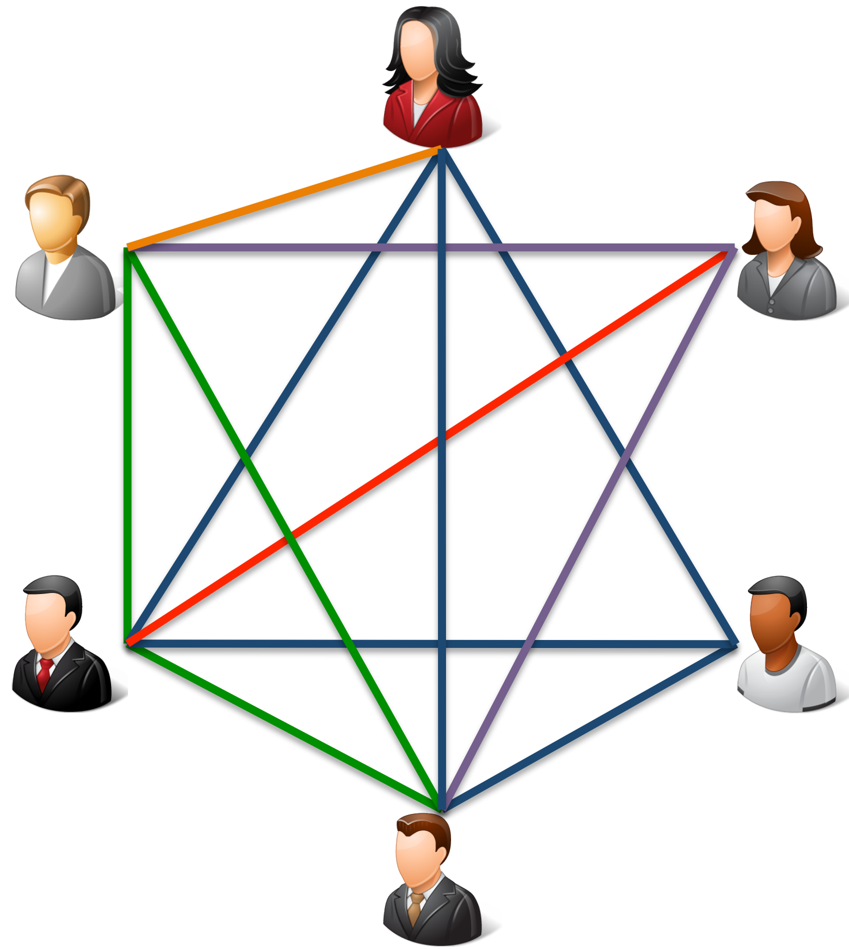


MOSP Graph

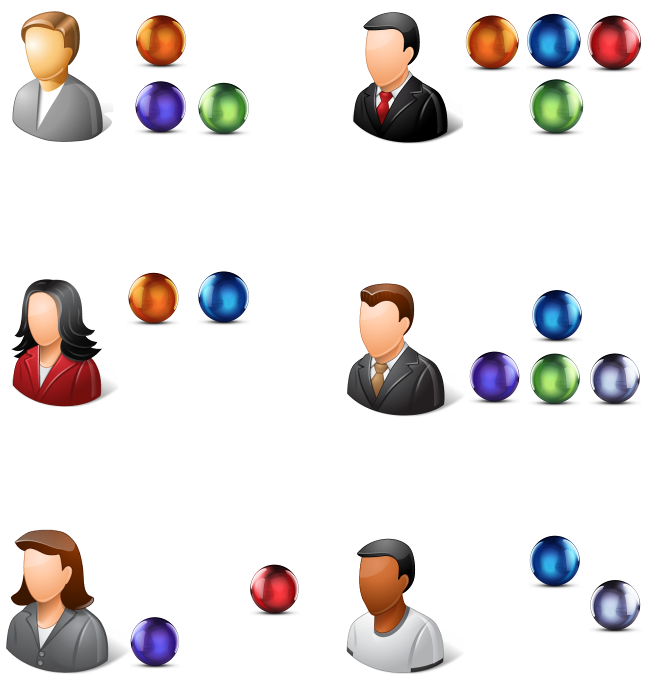


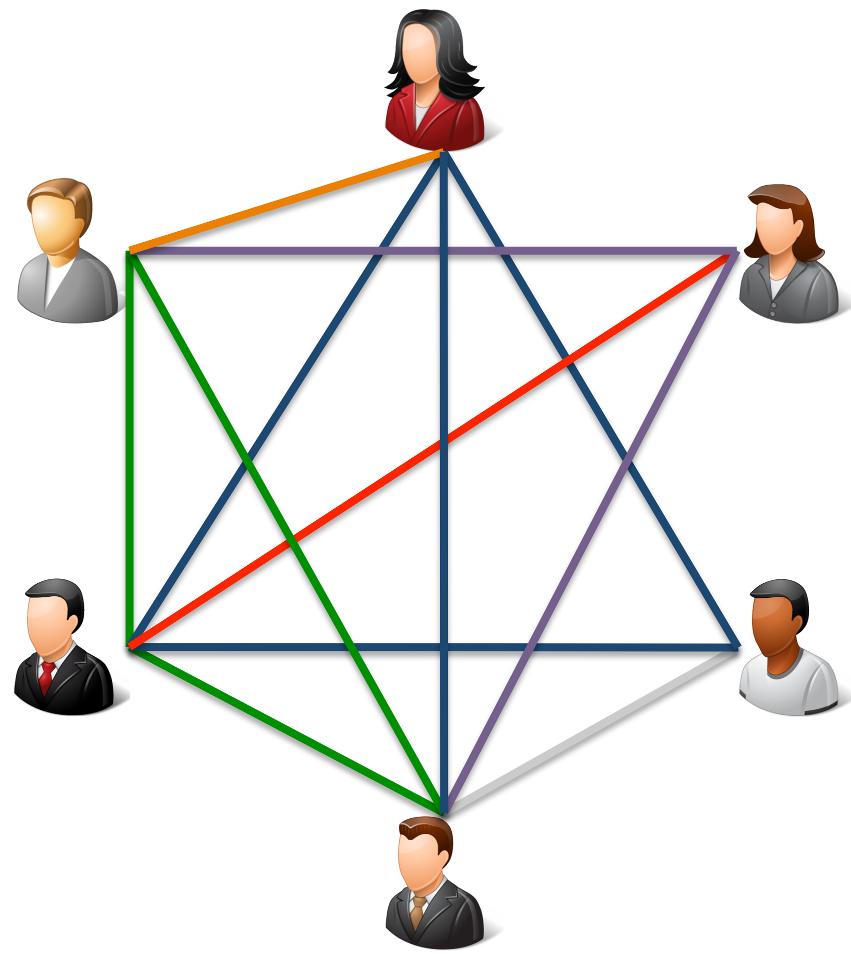
MOSP Graph





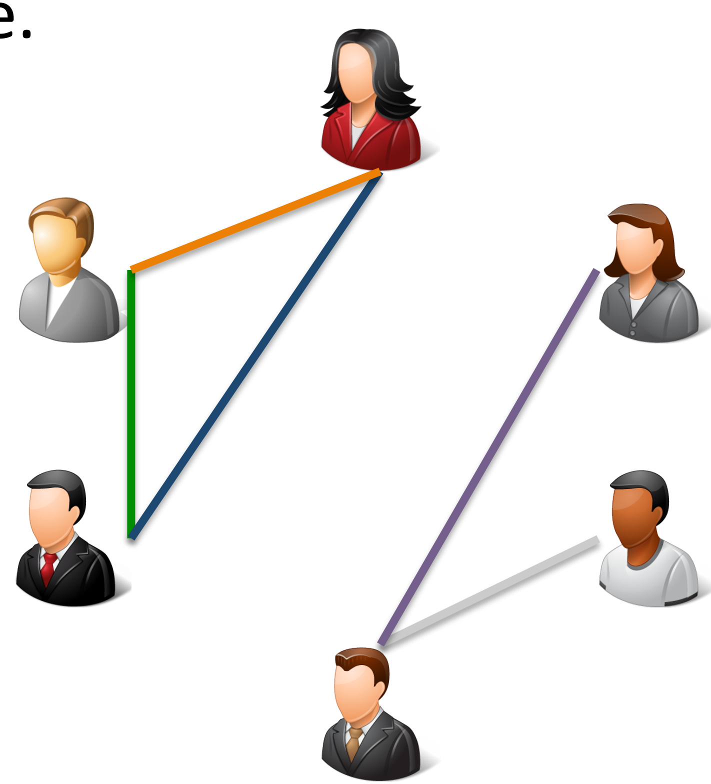
MOSP Graph





Preprocessing #1

- If the MOSP graph is disconnected, the problem is decomposable.



Preprocessing #2

- Let $c(p_i)$ determine the set of customers that ordered product p_i
  - If $c(p_j) \subseteq c(p_i)$, then p_i and p_j can be considered as one product.

	p1	p2	p3	p4	p5	p6
c1	1	0	0	1	1	0
c2	1	1	0	0	0	0
c3	0	0	1	1	0	0
c4	1	1	1	0	1	0
c5	0	1	0	1	1	1
c6	0	1	0	0	0	1

	p1	p2p6	p3	p4	p5
c1	1	0	0	1	1
c2	1	1	0	0	0
c3	0	0	1	1	0
c4	1	1	1	0	1
c5	0	1	0	1	1
c6	0	1	0	0	0

Breadth-First Search

- The MOSP resembles the *Matrix Bandwidth Minimization Problem* (MBM)
  - The MBM problem aims to find a permutation of rows and columns which keeps the nonzero elements of a matrix as close as possible to the main diagonal.

$$\begin{bmatrix} 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \Longrightarrow \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix}$$

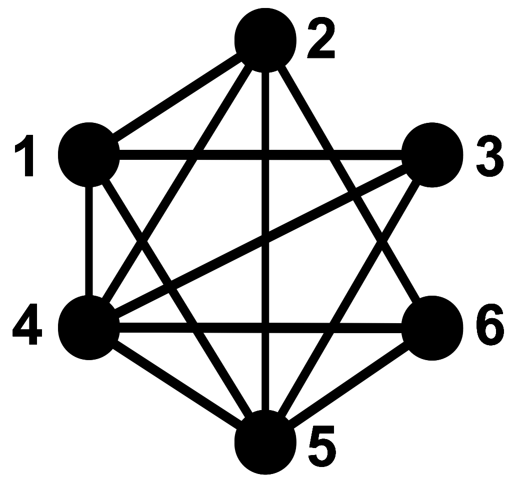
Breadth-First Search

- The *Cuthill-Mckee* (1969) heuristic for MBM explores a corresponding graph by Breadth-First Search (BFS):
  - Choice of lower degree nodes
    - Ties are broken in favor of the lower index node;
  - The sequence of the search determines the permutation of rows and columns.

Breadth-First Search

- The BFS has never been applied to the MOSP solution
  - MOSP instances may not be sparse, symmetric or square, as MBM matrices;
  - The band structure is not a required condition.
- However, when applied to the MOSP, it generates good results.

Breadth-First Search



- Not examined
- Examined
- All neighbors examined

Products Sequencing

- After sequencing the nodes (orders), we obtain the products permutation:
  - The orders are analyzed using LIFO policy;
  - Each ordered product is inserted in the solution using LIFO policy.

Products Sequencing

- Q={3, 1, 4, 5, 2, **6**}

	p1	p2p6	p3	p4	p5
c1	1	0	0	1	1
c2	1	1	0	0	0
c3	0	0	1	1	0
c4	1	1	1	0	1
c5	0	1	0	1	1
c6	0	**1**	0	0	0

	p2	p6
c1	0	0
c2	1	0
c3	0	0
c4	1	0
c5	1	1
c6	**1**	**1**

Products Sequencing

- Q={3, 1, 4, 5, **2**, 6}

	p1	p2p6	p3	p4	p5
c1	1	0	0	1	1
c2	**1**	**1**	0	0	0
c3	0	0	1	1	0
c4	1	1	1	0	1
c5	0	1	0	1	1
c6	0	1	0	0	0

	p1	p2	p6
c1	1	0	0
c2	**1**	1	0
c3	0	0	0
c4	1	1	0
c5	0	1	1
c6	0	1	1

Products Sequencing

- Q={3, 1, 4, **5**, 2, 6}

	p1	p2p6	p3	p4	p5
c1	1	0	0	1	1
c2	1	1	0	0	0
c3	0	0	1	1	0
c4	1	1	1	0	1
c5	0	**1**	0	**1**	**1**
c6	0	1	0	0	0

	p4	p5	p1	p2	p6
c1	1	1	1	0	0
c2	0	0	1	1	0
c3	1	0	0	0	0
c4	0	1	1	1	0
c5	**1**	**1**	0	1	1
c6	0	0	0	1	1

Products Sequencing

- Q={3, 1, **4**, 5, 2, 6}

	p1	p2p6	p3	p4	p5
c1	1	0	0	1	1
c2	1	1	0	0	0
c3	0	0	1	1	0
c4	**1**	**1**	**1**	0	**1**
c5	0	1	0	1	1
c6	0	1	0	0	0

	p3	p4	p5	p1	p2	p6
c1	0	1	1	1	0	0
c2	0	0	0	1	1	0
c3	1	1	0	0	0	0
c4	**1**	0	1	1	1	0
c5	0	1	1	0	1	1
c6	0	0	0	0	1	1

Products Sequencing

Manufacturing Sequence

Stacks

	p3	p4	p5	p1	p2	p6
c1		1	1	1		
c2				1	1	
c3	1	1				
c4	1	--	1	1	1	
c5		1	1	--	1	1
c6					1	1

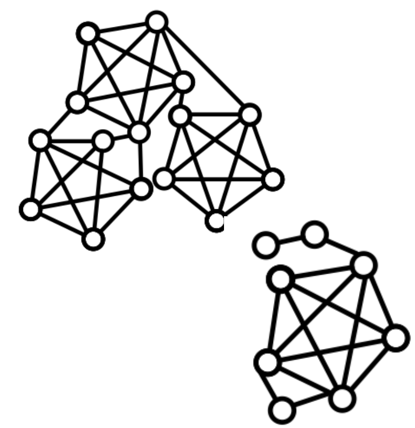
Max Open Stacks: 4

Breadth-First Search

- Breadth-First Search features:
  - Low degree nodes are not the problem's bottleneck
    - Sequenced first.
  - Clique's and high degree nodes tend to be sequenced contiguously;
  - Preprocessing #1 is inherent;
  - Computational complexity;
  - Ease of implementation.

Improvement Rules

- Special topologies of the MOSP graph cause BFS to generate errors:
  – Cliques loosely connected;
  – A dominant clique with a few nodes in its neighborhood.



- Improvement rules:
  1. Close inactive open stacks, by anticipating its product's manufacturing;
  2. Delay the opening of new stacks, by postponing its product’s manufacturing.

Improvement Rule #1

Manufacturing Sequence

Stacks	p4	p5	p2p6	p1	p3
c1	1	1	--	1	
c2			1	1	
c3	1	--	--	--	1
c4		1	1	1	1
c5	1	1	1		
c6			1		

Max Open Stacks: 6

Manufacturing Sequence

Stacks	p4	p5	p1	p2p6	p3
c1	1	1	1		
c2			1	1	
c3	1	--	--	--	1
c4		1	1	1	1
c5	1	1	--	1	
c6				1	

Max Open Stacks: 5

Improvement Rule #2

Manufacturing Sequence

Stacks	p5	p3	p1	p2p6	p4
c1	1	--	1	--	1
c2			1	1	
c3		1	--	--	1
c4	1	1	1	1	
c5	1	--	--	1	1
c6				1	

Max Open Stacks: 6

Manufacturing Sequence

Stacks	p5	p1	p2p6	p3	p4
c1	1	1	--	--	1
c2		1	1		
c3				1	1
c4	1	1	1	1	
c5	1	--	1	--	1
c6			1		

Max Open Stacks: 5

THE OR SOCIETY

Data sets

Computational Environment

COMPUTATIONAL EXPERIMENTS

Datasets

- First Constraint Modeling Challenge (2005)
  - 5,806 smaller instances;
  - Decomposable instances;
  - Polynomial topologies of MOSP graphs.
- Harder Instances (2009)
  - 200 larger instances;
  - No decomposable instances;
  - No polynomial topologies of MOSP graphs.

Computational Experiments

- Intel i5 Quad Core 3.2 GHz processor;
- 16 GB RAM;
- Ubuntu 12.4.1;
- No optimization options;
- Chu and Stuckey (2009) original code, compiled and run as recommended
  - MOSP state-of-the-art exact method.
- Implementation of Becceneri, Yanasse and Soma (2004) as originally described
  - MOSP state-of-the-art heuristic.

THE OR SOCIETY

Dataset #1

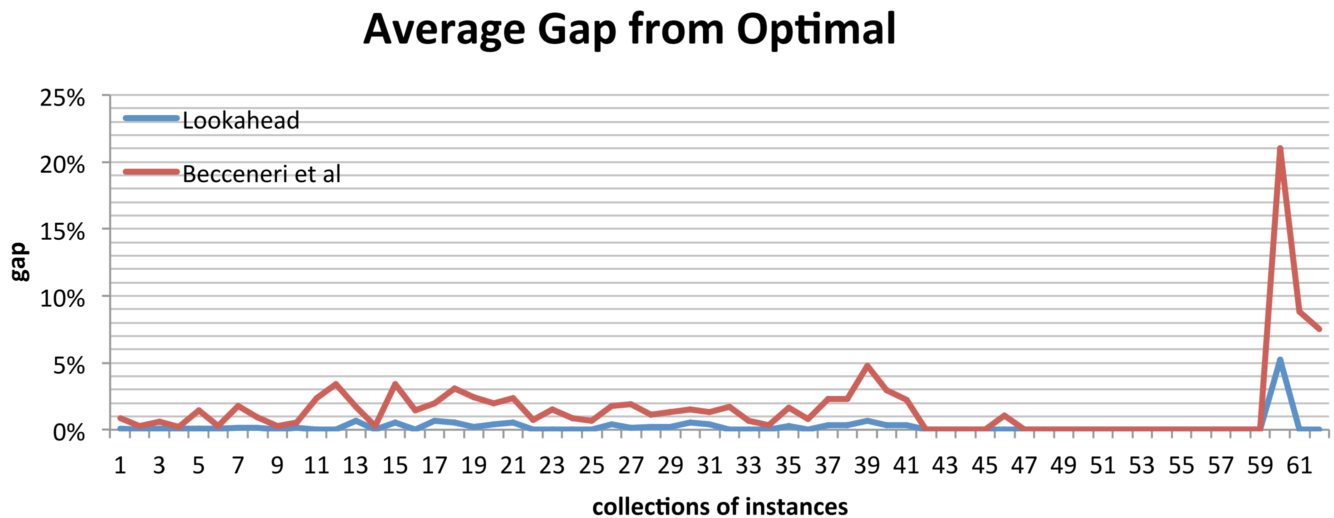
- Solutions

Method	Lookahead	Becceneri et al.
Best solutions	832 (14%)	41 (0.71%)
Optimal solutions	5,644 (97.21%)	4,889 (84.21%)
Max error from optimal	2 stacks	8 stacks
Gap from Optimal	0.18%	1.32%

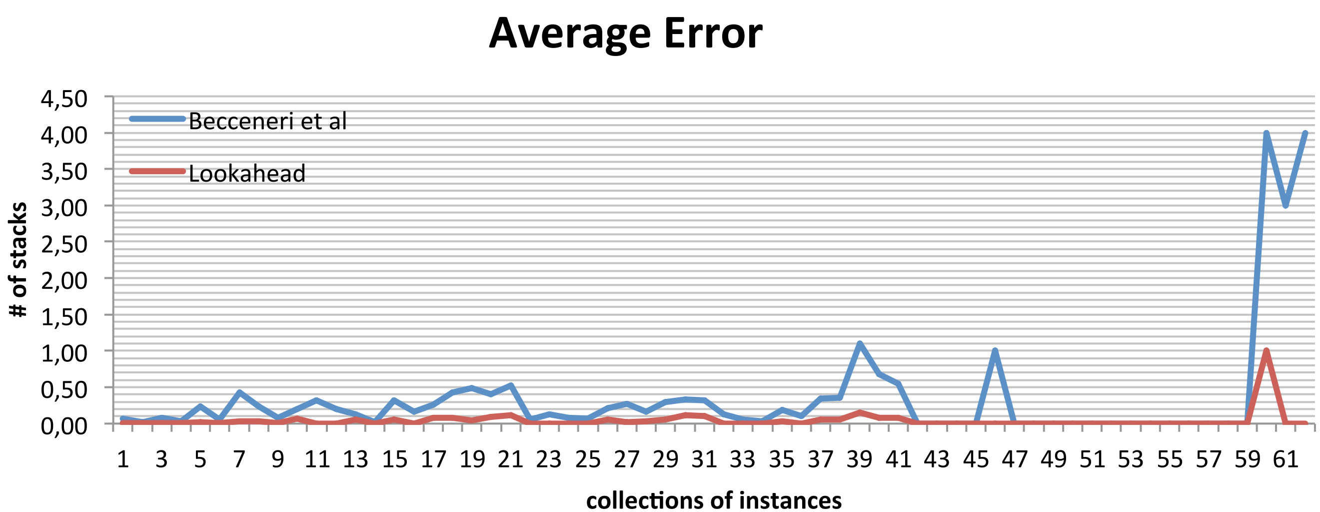
- Running times (ms)

Method	Min	Mean	Max
Chu and Stuckey	0.00	1.65	6,865.00
Lookahead	0.00	29.72	1,424.00
Becceneri et al.	0.00	0.02	24.00

Dataset #1



Average Gap from Optimal
25%
20%
15%
10%
5%
0%
gap
Lookahead
Becceneri et al
1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47 49 51 53 55 57 59 61
collections of instances



Average Error
4,50
4,00
3,50
3,00
2,50
2,00
1,50
1,00
0,50
0,00
# of stacks
Becceneri et al
Lookahead
1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47 49 51 53 55 57 59 61
collections of instances

Dataset #2

- Solutions

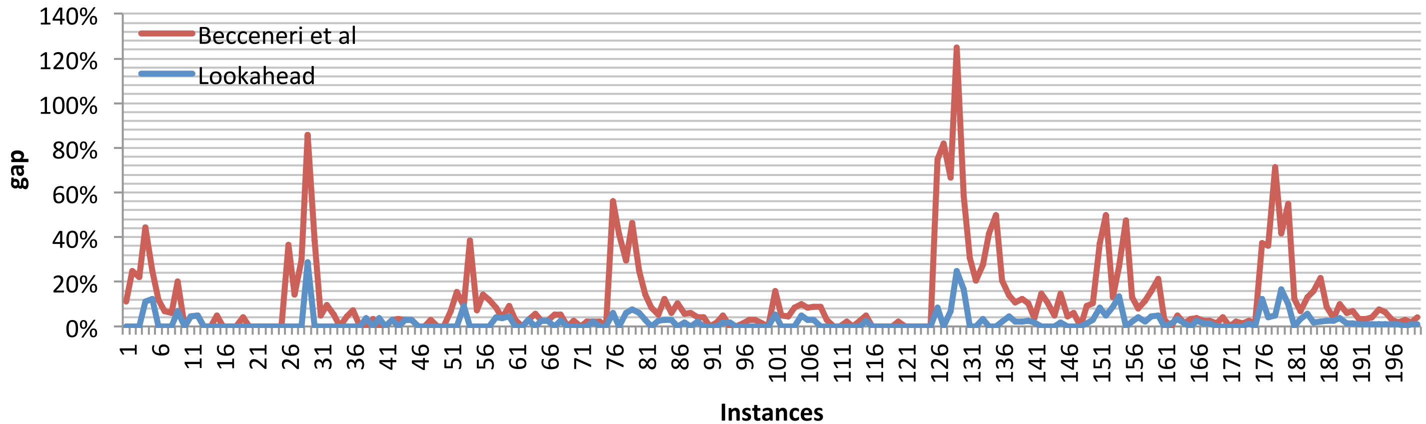
Method	Lookahead	Becceneri et al.
Best solutions	144 (72%)	4 (2%)
Optimal solutions	110 (55%)	44 (22%)
Max error from optimal	4 stacks	15 stacks
Gap from Optimal	1.41%	7.27%

- Running times (ms)

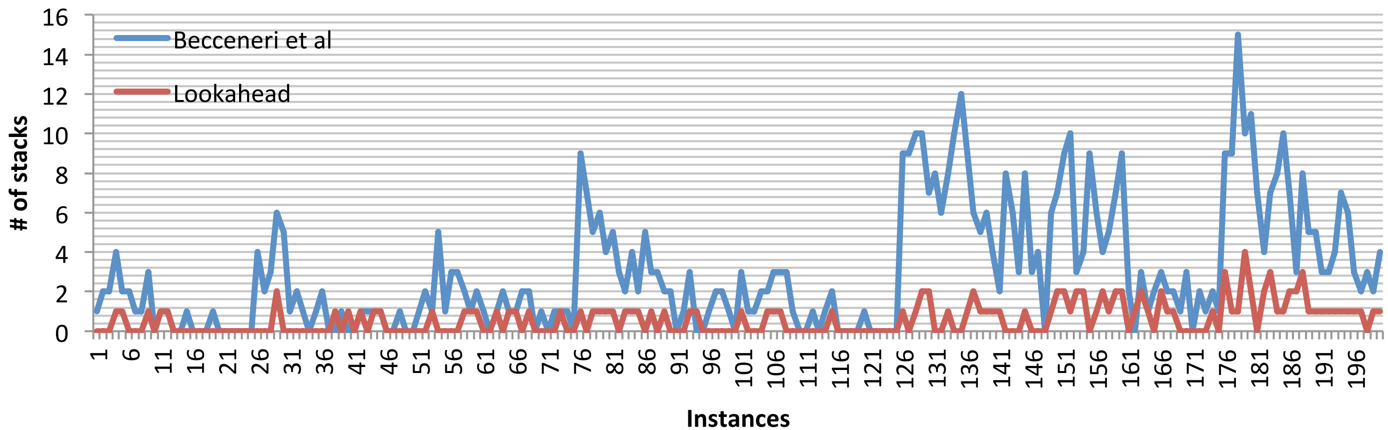
Method	Min	Mean	Max
Chu and Stuckey	0.00	12,851.00	945,151.00
Lookahead	0.00	1,587.00	15,220.00
Becceneri et al.	0.00	5.34	20.00

Dataset #2

Gap from Optimal



Error



THE OR SOCIETY

SUMMARY

Summary

- A novel approach to MOSP;
- $O(p^3)$ heuristic, where p denotes the number of products
  - Outperforms the state-of-the art heuristic in solution quality
    - Smaller gaps from optimal;
    - Robust - smaller errors;
    - Higher index of optimal solutions.
  - Fast.
- Can be used to generate good upper bounds;
- Can be used directly to solve MOSP and equivalent problems.

Acknowledgements

- Prof. Geoffrey Chu (University of Melbourne);
- This work was funded by the State of São Paulo Research Foundation - FAPESP, process 2009/51831-9 (first author).

Questions?

THANK YOU