A Preference-Based-Routing Example
Solved by Linear Programming with Excel Solver

From Soup to Nuts

A Small 10-Agent Example

Ward Whitt, May 26, 2005

A small example solved by Excel with Solver (basic LP).

Suppose that the Contact Center is set up to handle the following interactions:

Variable	VariableTypes
Language	French, English
Purpose	Service, Technical

Here are the 4 feasible CallType combinations:

CallType	Language	Purpose	Symbol
1	English	Technical	ET
2	English	Service	ES
3	French	Technical	FT
4	French	Service	FS

The goal is to assign agents to priority pairs.
There are 4*3 = 12 possible ordered priority pairs. We consider only 8.
(We assume in advance that the other 4 will not be assigned.)

Assume that there are n = 10 agents.

See the very end for a summary of model input and output.

The Skill Matrix - a 10 x 4 matrix S = (Sij)

First Representation: 5 bilingual, 2 English only, and 3 French only; all can do both T and S

		Language		***Purpose***		
		E	F	T	S	<u>Row sums</u> (Skill Scores)
	A1	1	1	1	1	4
	A2	1	1	1	1	4
	A3	1	1	1	1	4
	A4	1	1	1	1	4
	A5	1	1	1	1	4
Agents	A6	1	0	1	1	3
	A7	1	0	1	1	3
	A8	0	1	1	1	3
	A9	0	1	1	1	3
	A10	0	1	1	1	3
	<u>column sum</u>	7	8	10	10	

Second (desired) representation: 10 x 4 -- agents x call type

		Call Type				
		1 = ET	2 = ES	3 = FT	4 = FS	<u>Row sums</u> (Skill Scores)
	A1	1	1	1	1	4
	A2	1	1	1	1	4
	A3	1	1	1	1	4
	A4	1	1	1	1	4
	A5	1	1	1	1	4
Agents	A6	1	1	0	0	2
	A7	1	1	0	0	2
	A8	0	0	1	1	2
	A9	0	0	1	1	2
	A10	0	0	1	1	2
	<u>column sum</u>	7	7	8	8	

Preference Scheme

Score	Interpretation	Assigned By
3	High Preference	Agent
2	Medium Prefere	Agent
1	Low Preference	Agent
0	Agent lacks skill	System

For a particular period of time (e.g., half hour or shift), the following preferences have been expressed:

A multiplier equal to number of skills divided by 4 is used for each agent
Agent preferences assumed for illustrative purposes.

		multipliers		Agents directly express ***call type***				Modified by multiplier ***call type***			
		no. of skills	multiplier	1 = ET	2 = ES	3 = FT	4 = FS	1 = ET	2 = ES	3 = FT	4 = FS
	A1	4	1	3	2	1	1	3	2	1	1
	A2	4	1	3	1	3	1	3	1	3	1
	A3	4	1	1	3	1	2	1	3	1	2
	A4	4	1	3	3	1	1	3	3	1	1
	A5	4	1	2	1	3	1	2	1	3	1
Agents	A6	2	0.5	3	2	0	0	1.5	1	0	0
	A7	2	0.5	2	3	0	0	1	1.5	0	0
	A8	2	0.5	0	0	3	2	0	0	1.5	1
	A9	2	0.5	0	0	2	3	0	0	1	1.5
	A10	2	0.5	0	0	2	3	0	0	1	1.5

Agent Preference matrix A = (Aij) -- here the same as R = (Rij)

(Taken from last four columns above)

column sums

		call type			
		1 = ET	2 = ES	3 = FT	4 = FS
	A1	3	2	1	1
	A2	3	1	3	1
	A3	1	3	1	2
	A4	3	3	1	1
	A5	2	1	3	1
Agents	A6	1.5	1	0	0
	A7	1	1.5	0	0
	A8	0	0	1.5	1
	A9	0	0	1	1.5
	A10	0	0	1	1.5

column sums

Shift to Priority Pairs

Priority Pair	Primary	Secondary
1	ET	ES
2	ET	FT
3	ES	ET
4	ES	FS
5	FT	ET
6	FT	FS
7	FS	ES
8	FS	FT

Preliminary Rewards for priority pairs: rr(I,j,k) = p*Rij + (1-p)*Rik (Does not yet account for skills)

probability p= 0.7

		1=ET-ES	2=ET-FT	3=ES-ET	4=ES-FS	5=FT-ET	6=FT-FS	7=FS-ES	8=FS-FT	row sur	rowmax
	A1	2.7	2.4	1.7	1.7	1.6	1	1.3	1	13.4	2.7
	A2	2.4	3	1	1	3	2.4	1	1.6	15.4	3
	A3	1.6	1	2.7	2.7	1	1.3	2.3	1.7	14.3	2.7
	A4	3	2.4	2.4	2.4	1.6	1	1.6	1	15.4	3
	A5	1.7	2.3	1	1	2.7	2.4	1	1.6	13.7	2.7
Agents	A6	1.35	1.05	0.7	0.7	0.45	0	0.3	0	4.55	1.35
	A7	1.15	0.7	1.05	1.05	0.3	0	0.45	0	4.7	1.15
	A8	0	0.45	0.3	0.3	1.05	1.35	0.7	1.15	5.3	1.35
	A9	0	0.3	0.45	0.45	0.7	1.15	1.05	1.35	5.45	1.35
	A10	0	0.3	0.45	0.45	0.7	1.15	1.05	1.35	5.45	1.35
	column sum	13.9	13.9	11.75	11.75	13.1	11.75	10.75	10.75	97.65	20.65
									average	1.221	

Skills for priority pairs -- s(I,j,k)

		1=ET-ES	2=ET-FT	3=ES-ET	4=ES-FS	5=FT-ET	6=FT-FS	7=FS-ES	8=FS-FT	row sums
	A1	1	1	1	1	1	1	1	1	8
	A2	1	1	1	1	1	1	1	1	8
	A3	1	1	1	1	1	1	1	1	8
	A4	1	1	1	1	1	1	1	1	8
	A5	1	1	1	1	1	1	1	1	8
Agents	A6	1	0	1	0	0	0	0	0	2
	A7	1	0	1	0	0	0	0	0	2
	A8	0	0	0	0	0	1	0	1	2
	A9	0	0	0	0	0	1	0	1	2
	A10	0	0	0	0	0	1	0	1	2
	column sum	7	5	7	5	5	8	5	8	50

Final Rewards for priority pairs: r(I,j,k) = rr(I,j,k)*s(I,j,k) -- rewards above except when agent does not have skill

		1=ET-ES	2=ET-FT	3=ES-ET	4=ES-FS	5=FT-ET	6=FT-FS	7=FS-ES	8=FS-FT	row sums
	A1	2.7	2.4	1.7	1.7	1.6	1	1.3	1	13.4
	A2	2.4	3	1	1	3	2.4	1	1.6	15.4
	A3	1.6	1	2.7	2.7	1	1.3	2.3	1.7	14.3
	A4	3	2.4	2.4	2.4	1.6	1	1.6	1	15.4
	A5	1.7	2.3	1	1	2.7	2.4	1	1.6	13.7
Agents	A6	1.35	0	0.7	0	0	0	0	0	2.05
	A7	1.15	0	1.05	0	0	0	0	0	2.2
	A8	0	0	0	0	0	1.35	0	1.15	2.5
	A9	0	0	0	0	0	1.15	0	1.35	2.5
	A10	0	0	0	0	0	1.15	0	1.35	2.5
	column sum	13.9	11.1	10.55	8.8	9.9	11.75	7.2	10.75	83.95
									average	1.049

Reward matrix r = (rij) -- a 10 x 8 matrix above

We henceforth let j index the priority pairs, as in the new algorithm paper.

The Optimization Problem

There are n agents. Suppose here n = 10.

There are m pairs (priority pairs). Here m = 8.

We are given score (reward) rij = r(i,j) for agent i to handle pair j.

The Goal: maximize the total reward,
i.e., maximize the sum of xij*rij over i and j,
where xij = 1 means agent i is assigned pair j, while
xij = 0 otherwise

Subject to constraints:

Each agent is given one pair assignment.
Specified number of priority pairs assigned.

Priority Pair Requirements

Priority Pair	Primary	Secondary	Required	Pair
1	ET	ES	2	12
2	ET	FT	1	13
3	ES	ET	1	21
4	ES	FS	1	24
5	FT	ET	1	31
6	FT	FS	1	34
7	FS	ES	2	42
8	FS	FT	1	43

The sum of the requirements equals the number of agents.

Agents	10

We assume that the requirements for priority pairs are given.

The staffing function is critically important, a prerequisite.

We here are only concerned with generating priority-pair assignments.

The routing based on priority pairs is assumed given as well.

Rewards below are rij = r(i,j) - for agent i handling priority ordered pair j

The Arc-Flow Constraints

S = source T = sink, terminal

From	To	Flow	<=	capacity	reward rate rij	reward for this flow
S	Agent1	1	<=	1	0	0
S	Agent2	1	<=	1	0	0
S	Agent3	1	<=	1	0	0
S	Agent4	1	<=	1	0	0
S	Agent5	1	<=	1	0	0
S	Agent6	1	<=	1	0	0
S	Agent7	1	<=	1	0	0
S	Agent8	1	<=	1	0	0
S	Agent9	1	<=	1	0	0
S	Agent10	1	<=	1	0	0
Pair12	T	2	<=	2	0	0
Pair13	T	1	<=	1	0	0
Pair21	T	1	<=	1	0	0
Pair24	T	1	<=	1	0	0
Pair31	T	1	<=	1	0	0
Pair34	T	1	<=	1	0	0
Pair42	T	2	<=	2	0	0

Pair43	T	1	<=	1	0	0
Agent1	Pair12	1	<=	20	2.7	2.7
Agent1	Pair13	0	<=	20	2.4	0
Agent1	Pair21	0	<=	20	1.7	0
Agent1	Pair24	0	<=	20	1.7	0
Agent1	Pair31	0	<=	20	1.6	0
Agent1	Pair34	0	<=	20	1	0
Agent1	Pair42	0	<=	20	1.3	0
Agent1	Pair43	0	<=	20	1	0
Agent2	Pair12	0	<=	20	2.4	0
Agent2	Pair13	1	<=	20	3	3
Agent2	Pair21	0	<=	20	1	0
Agent2	Pair24	0	<=	20	1	0
Agent2	Pair31	0	<=	20	3	0
Agent2	Pair34	0	<=	20	2.4	0
Agent2	Pair42	0	<=	20	1	0
Agent2	Pair43	0	<=	20	1.6	0
Agent3	Pair12	0	<=	20	1.6	0
Agent3	Pair13	0	<=	20	1	0
Agent3	Pair21	0	<=	20	2.7	0
Agent3	Pair24	0	<=	20	2.7	0
Agent3	Pair31	0	<=	20	1	0
Agent3	Pair34	0	<=	20	1.3	0
Agent3	Pair42	1	<=	20	2.3	2.3
Agent3	Pair43	0	<=	20	1.7	0
Agent4	Pair12	0	<=	20	3	0
Agent4	Pair13	0	<=	20	2.4	0
Agent4	Pair21	0	<=	20	2.4	0
Agent4	Pair24	1	<=	20	2.4	2.4
Agent4	Pair31	0	<=	20	1.6	0
Agent4	Pair34	0	<=	20	1	0
Agent4	Pair42	0	<=	20	1.6	0
Agent4	Pair43	0	<=	20	1	0
Agent5	Pair12	0	<=	20	1.7	0
Agent5	Pair13	0	<=	20	2.3	0
Agent5	Pair21	0	<=	20	1	0
Agent5	Pair24	0	<=	20	1	0

Agent5	Pair31	1	<=	20	2.7	2.7
Agent5	Pair34	0	<=	20	2.4	0
Agent5	Pair42	0	<=	20	1	0
Agent5	Pair43	0	<=	20	1.6	0
Agent6	Pair12	1	<=	20	1.35	1.35
Agent6	Pair13	0	<=	20	0	0
Agent6	Pair21	0	<=	20	0.7	0
Agent6	Pair24	0	<=	20	0	0
Agent6	Pair31	0	<=	20	0	0
Agent6	Pair34	0	<=	20	0	0
Agent6	Pair42	0	<=	20	0	0
Agent6	Pair43	0	<=	20	0	0
Agent7	Pair12	0	<=	20	1.15	0
Agent7	Pair13	0	<=	20	0	0
Agent7	Pair21	1	<=	20	1.05	1.05
Agent7	Pair24	0	<=	20	0	0
Agent7	Pair31	0	<=	20	0	0
Agent7	Pair34	0	<=	20	0	0
Agent7	Pair42	0	<=	20	0	0
Agent7	Pair43	0	<=	20	0	0
Agent8	Pair12	0	<=	20	0	0
Agent8	Pair13	0	<=	20	0	0
Agent8	Pair21	0	<=	20	0	0
Agent8	Pair24	0	<=	20	0	0
Agent8	Pair31	0	<=	20	0	0
Agent8	Pair34	1	<=	20	1.35	1.35
Agent8	Pair42	0	<=	20	0	0
Agent8	Pair43	0	<=	20	1.15	0
Agent9	Pair12	0	<=	20	0	0
Agent9	Pair13	0	<=	20	0	0
Agent9	Pair21	0	<=	20	0	0
Agent9	Pair24	0	<=	20	0	0
Agent9	Pair31	0	<=	20	0	0
Agent9	Pair34	0	<=	20	1.15	0
Agent9	Pair42	1	<=	20	0	0
Agent9	Pair43	0	<=	20	1.35	0
Agent10	Pair12	0	<=	20	0	0

Agent10	Pair13	0	<=	20	0	0
Agent10	Pair21	0	<=	20	0	0
Agent10	Pair24	0	<=	20	0	0
Agent10	Pair31	0	<=	20	0	0
Agent10	Pair34	0	<=	20	1.15	0
Agent10	Pair42	0	<=	20	0	0
Agent10	Pair43	1	<=	20	1.35	1.35
Totals		10				18.2
Average						1.82

The number of constraints is 10+8+(10*8) = 98.
The first 10 constraints say each agent works.
The next 8 specify the priority pair assignments.
The last 8*10 = 80 relate to agent-pair assignments.

The Node Flow Constraints

Nodes	Flow in	Flow out	Net flow	Supply/Demand	
S	0	10	-10	equals	-10
T	10	0	10	equals	10
Agent1	1	1	0	equals	0
Agent2	1	1	0	equals	0
Agent3	1	1	0	equals	0
Agent4	1	1	0	equals	0
Agent5	1	1	0	equals	0
Agent6	1	1	0	equals	0
Agent7	1	1	0	equals	0
Agent8	1	1	0	equals	0
Agent9	1	1	0	equals	0
Agent10	1	1	0	equals	0
Pair12	2	2	0	equals	0
Pair13	1	1	0	equals	0
Pair21	1	1	0	equals	0
Pair24	1	1	0	equals	0
Pair31	1	1	0	equals	0
Pair34	1	1	0	equals	0

Pair42	2	2	0	equals	0
Pair43	1	1	0	equals	0

There are 20 node constraints.

Final Optimal Assignment

Agent1	Pair12
Agent2	Pair13
Agent3	Pair42
Agent4	Pair24
Agent5	Pair31
Agent6	Pair12
Agent7	Pair21
Agent8	Pair34
Agent9	Pair43
Agent10	Pair42

You need to round to the nearest integer.

What can be gained by optimizing?

By running Solver with the minimum, we see the worst case.
By running Solver with the maximum, we see the best case.

Best case = 18.2 total reward, = 1.82 per agent
Worst case = 5.0 total reward =0.50 per agent

Best case letting each agent do anything 20.65, 2.065 per agent
obtained by removing some constraints
Each agent picks j with largest Rij -- see cell n159

Summary of Solver Input Settings:

Set Target Cell $G$355 - total reward
Equal to Max
By changing cells $C$256:$C$353
Subject to Constraints
$C$256:$C$353 <= $E$256:$E$353
$C$256:$C$353 >= 0
$D$366 = -10
$D$367 = 10
$D$368:$D$385 = 0

Solver Options

Max Time 100 seconds
Iterations 1000
precision 0.0001
tolerance 5%
convergence = 0.0001

Assume Linear Model (want)
Assume nonnegative (redundant)

Estimates (not needed)

Can show iteration results to step through all steps.
But do not do that to instantly get to the solution.

Summary of Iteration History

for this example

Total number of iterations = 68

Iterations with total rewards achieved

Now presenting the final priority matrix P

First extracting assignments from cells C274 to C353:

					Agents						
		1	2	3	4	5	6	7	8	9	10	row sums
	12	1	0	0	0	0	1	0	0	0	0	2
	13	0	1	0	0	0	0	0	0	0	0	1
	21	0	0	0	0	0	0	1	0	0	0	1
Pairs	24	0	0	0	1	0	0	0	0	0	0	1
	31	0	0	0	0	1	0	0	0	0	0	1
	34	0	0	0	0	0	0	0	1	0	0	1
	42	0	0	1	0	0	0	0	0	1	0	2
	43	0	0	0	0	0	0	0	0	0	1	1
												0
	column sum	1	1	1	1	1	1	1	1	1	1	10

(Rounding done below.)

Now we give the Agent Pair Assignments - transpose of the matrix above

(I did not use the transpose array function, but I would next time.)
(Rounding done here.)

		Pairs								
		12	13	21	24	31	34	42	43	row sums
	1	1	0	0	0	0	0	0	0	1
	2	0	1	0	0	0	0	0	0	1
	3	0	0	0	0	0	0	1	0	1
	4	0	0	0	1	0	0	0	0	1
Agents	5	0	0	0	0	1	0	0	0	1
	6	1	0	0	0	0	0	0	0	1
	7	0	0	1	0	0	0	0	0	1
	8	0	0	0	0	0	1	0	0	1
	9	0	0	0	0	0	0	1	0	1
	10	0	0	0	0	0	0	0	1	1

Finally, here is the desired 10 x 8 priority matrix   P = (Pij)

		Call Types				Row sums
		1=ET	2=ES	3=FT	4=FS	
	1	3	2	0	0	5
	2	3	0	2	0	5
	3	0	2	0	3	5
	4	0	3	0	2	5
Agents	5	2	0	3	0	5
	6	3	2	0	0	5
	7	2	3	0	0	5
	8	0	0	3	2	5
	9	0	2	0	3	5
	10	0	0	2	3	5
column sums		13	14	10	13	

(We could replace some 0's with 1's if we wanted.)

Model Input

agent skills	C45:F54		
agent preferences:	E98:H107		
staffing requirements	E266:E273	(also D224:D231)	(D366 and D367 must be consistent.)
probabiity weight for r(i,j,k)	C145		

Main Model Output

final priority matrix	D508:G517

Intermediate Model Output

agent skill matrix	C62:F71
agent preference. matrix	C117:F126
reward matrix (by pairs)	C181:J190
arc flows (decision vars.)	C256:C353