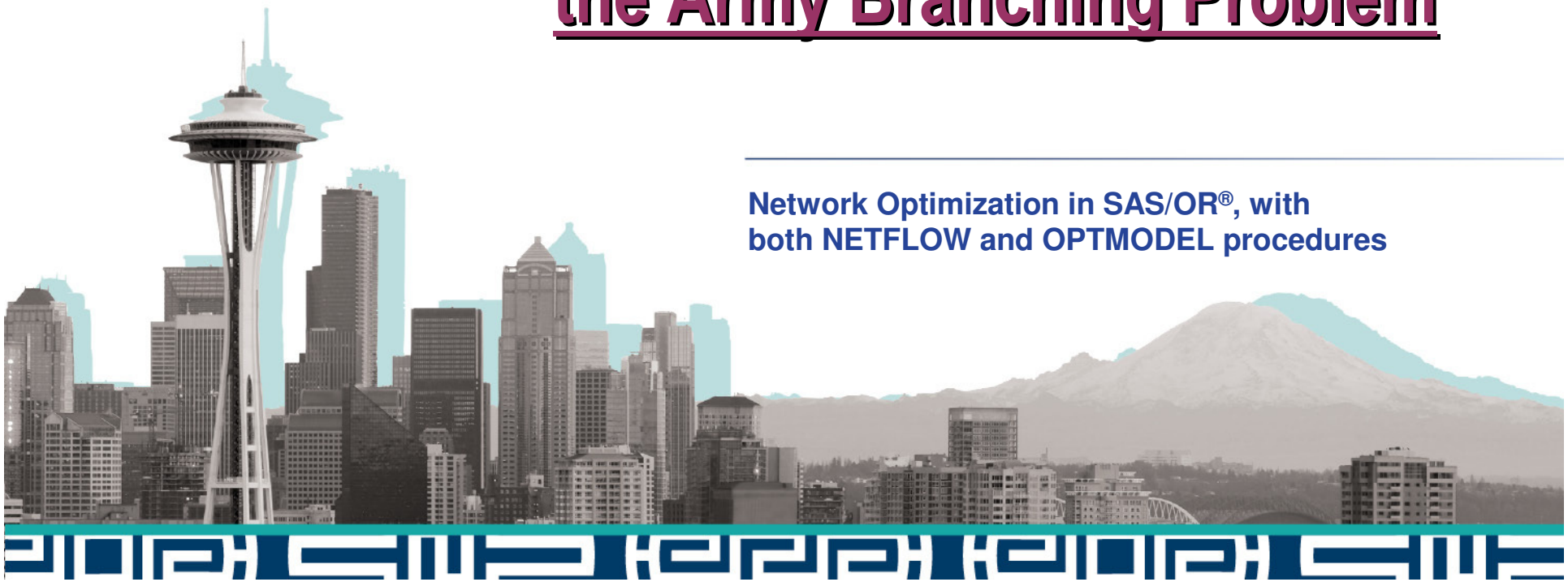
SAS GLOBAL FORUM

A Network Optimization Solution Using SAS/OR® Tools for the Department of the Army Branching Problem



Network Optimization in SAS/OR®, with both NETFLOW and OPTMODEL procedures

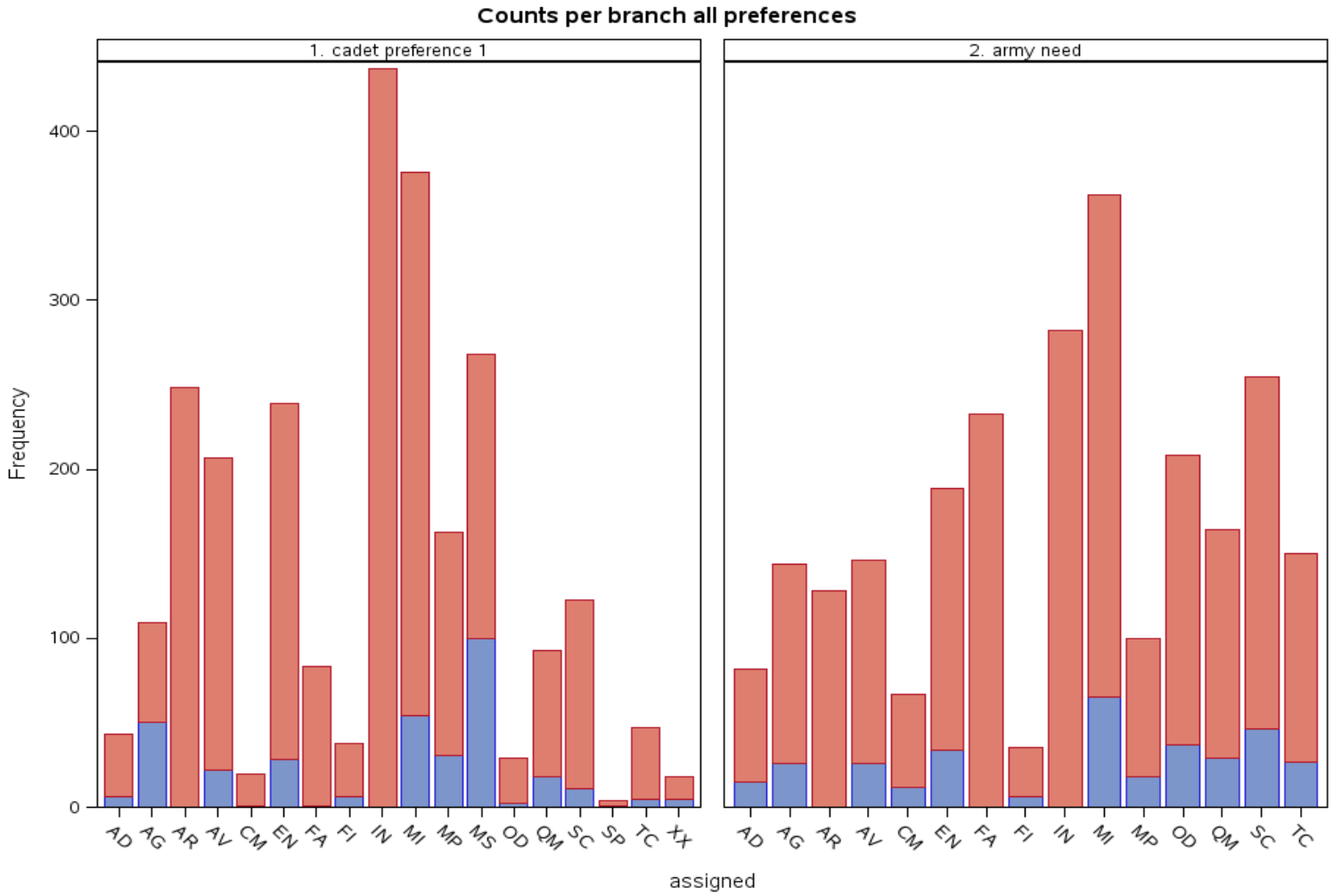
Presenter

- Michael C Grierson
  - Operations Research Analyst
  - The Department of the Army
    - Officer Personnel Management Directorate
    - Analytics to support Officer Planning and Assignments

The Problem and example

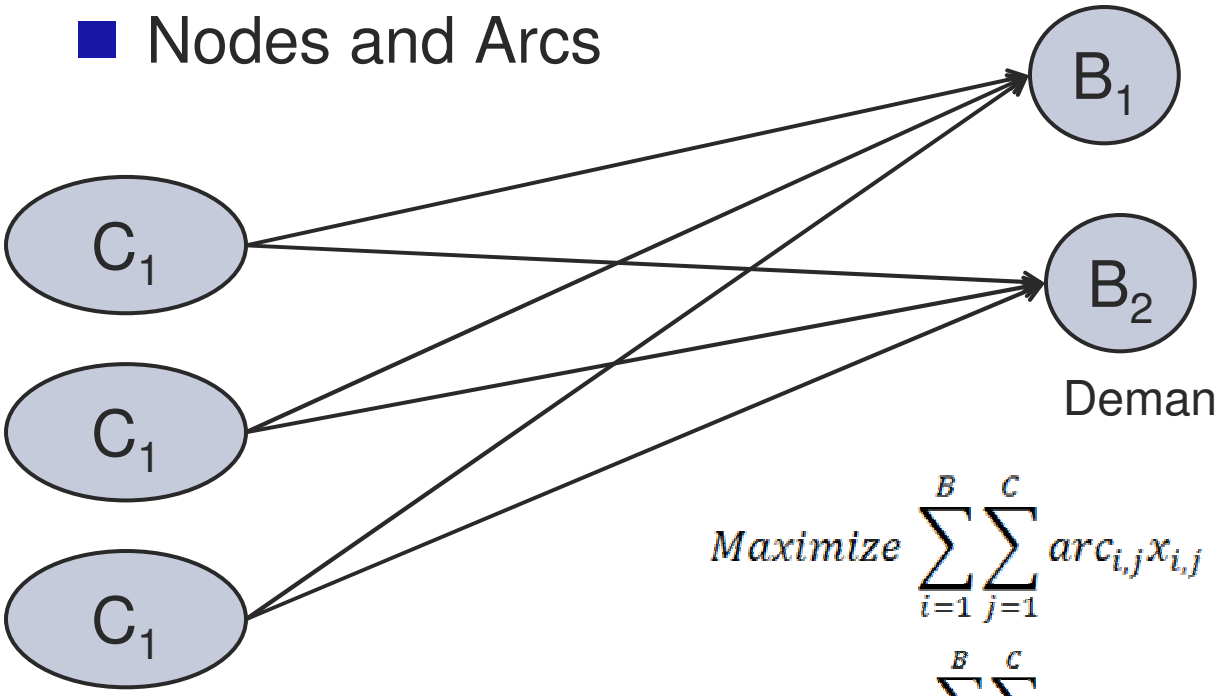
- Assign ROTC cadets to their initial Basic Branch
- Start with a 'supply' of 2545 ROTC cadets.
  - 5 branch preferences
  - gender
  - Order of Merit Score (OMS)
- Army has 2545 branch needs.
- **Objective** is to **maximize cadet satisfaction**
  - Satisfied if assignment is 1st, 2nd, or 3rd preference.
- **Subject to constraints of**
  - Army **basic branch 'demand'** to fill assignments.
  - **No Females in Combat Arms** (IN, AR, FA) branches
  - **Proportionally distribute** cadets by gender, and OMS.

1st Cadet Preference vs. Army Need



Network Models

- Nodes and Arcs



$$Maximize \sum_{i=1}^{B}\sum_{j=1}^{C} arc_{i,j}x_{i,j}$$

$$subject\ to \sum_{i=1}^{B}\sum_{j=1}^{C} x_{i,j} \geq B_i$$

$$subject\ to \sum_{j=1}^{C}\sum_{i=1}^{B} x_{i,j} = 1$$

Model Data (nodes)

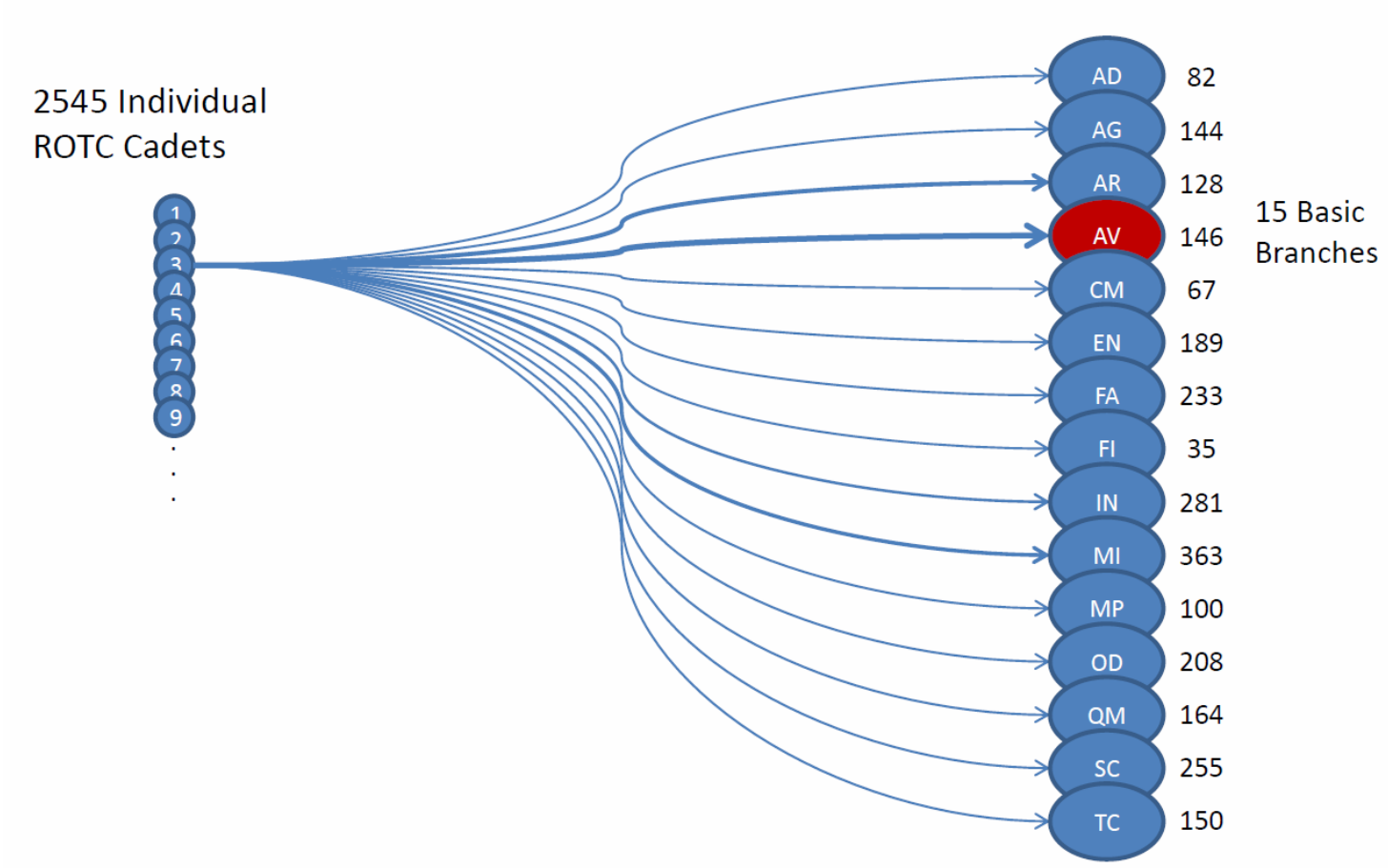
Cadets (Supply)

Number	OMS_Score	Sex	P1	P2	P3	P4	P5
1	97.21103	M	IN	EN	FA	MP	SC
2	96.85913	F	AV	AG	FI	MS	MI
3	96.69296	M	AV	MS	AR	MI	IN
4	96.5456	M	AV	IN	EN	FA	AR
5	96.22152	M	IN	MI	EN	MP	AR
6	95.98078	M	EN	MI	IN	AR	MS
	•						
	•						
	•						

Branches (Demand)

Branch Name	Branch Code	demand
Air Defense	AD	82
Adjutant General	AG	144
Armor	AR	128
Aviation	AV	146
Chemical	CM	67
Engineering	EN	189
Field Artillery	FA	233
Finance	FI	35
Infantry	IN	282
Military Intelligence	MI	362
Military Police	MP	100
Ordinance	OD	208
Quartermaster	QM	164
Signal Corp	SC	255
Transportation Corp	TC	150
	Total	2545

Network arcs, supply to demand



Scoring Arcs

- All arcs initially scored as 1
- Score then adjusted for preferences,
  - if 1st preference then that arc is given, +5
  - if 2nd then +4, if 3rd then +3,
  - if 4th then +2, if 5th then +1.
- Score then adjusted for OMS like so,
  - new Score = current Score/OMS ranking

NETFLOW Datasets (Nodes and Arcs)

node	_sd_
1	1
2	1
3	1
•	
•	
•	
2544	1
2545	1
ad	-82
ag	-144
ar	-128
av	-146
cm	-67
en	-189
fa	-233
fi	-35
in	-281
mi	-363
mp	-100
od	-208
qm	-164
sc	-255
tc	-150

Node and Arc data for cadet 3 (with scores called _cost_)

Obs	_from_	_cost_	_name_	_to_	_capac_	_lo_
31	3	0.3333	3_ad	ad	1	.
32	3	0.3333	3_ag	ag	1	.
33	3	0.3333	3_cm	cm	1	.
34	3	0.3333	3_en	en	1	.
35	3	0.3333	3_fa	fa	1	.
36	3	0.3333	3_fi	fi	1	.
37	3	0.3333	3_tc	tc	1	.
38	3	0.3333	3_mp	mp	1	.
39	3	0.3333	3_od	od	1	.
40	3	0.3333	3_qm	qm	1	.
41	3	0.3333	3_sc	sc	1	.
42	3	0.6667	3_in	in	1	.
43	3	1	3_mi	mi	1	.
44	3	1.3333	3_ar	ar	1	.
45	3	2	3_av	av	1	.

NETFLOW

```
proc netflow maximize
    nodedata=node_data
    arcdata=arc_data
    arcout= sol;
    reset maxit1 = 30000;
run;
```

Initial Solution

Cadet Satisfaction

(1st, 2nd, or 3rd preference matched)

choice	COUNT	PERCENT	CUM
1st	1572	61.768	61.768
2nd	640	25.147	86.916
3rd	253	9.941	96.857
4th	41	1.611	98.468
5th	14	0.550	99.018
other	25	0.982	100.000

Demand (Goal)

BR	ASSIGNED	GOAL
AD	82	82
AG	144	144
AR	128	128
AV	146	146
CM	67	67
EN	189	189
FA	233	233
FI	35	35
IN	281	281
MI	363	363
MP	100	100
OD	208	208
QM	164	164
SC	255	255
TC	150	150

from	_to_	_cost_	_capac_	_lo_	_name_	_SUPPLY_	_DEMAND_	_FLOW_	_FCOST_	_RCOST_	_ANUMB_	_TNUMB_	_STATUS_	
3	ad	0.33333	1	0	3_ad	1	82	0	0	-1.66	11836	33	LOWERBD	NONBASIC
3	ag	0.33333	1	0	3_ag	1	144	0	0	-1.66	13594	33	LOWERBD	NONBASIC
3	ar	1	1	0	3_ar	1	128	0	0	-1	34688	33	LOWERBD	NONBASIC
3	av	2	1	0	3_av	1	146	1	2	.	36445	33	KEY_ARC	BASIC
3	cm	0.33333	1	0	3_cm	1	67	0	0	-1.66	15352	33	LOWERBD	NONBASIC
3	en	0.33333	1	0	3_en	1	189	0	0	-1.66	17110	33	LOWERBD	NONBASIC
3	fa	0.33333	1	0	3_fa	1	233	0	0	-1.66	18868	33	LOWERBD	NONBASIC
3	fi	0.33333	1	0	3_fi	1	35	0	0	-1.66	20625	33	LOWERBD	NONBASIC
3	in	0.33333	1	0	3_in	1	281	0	0	-1.67	22383	33	LOWERBD	NONBASIC
3	mi	0.66667	1	0	3_mi	1	363	0	0	-1.33	32930	33	LOWERBD	NONBASIC
3	mp	0.33333	1	0	3_mp	1	100	0	0	-1.66	24140	33	LOWERBD	NONBASIC
3	od	0.33333	1	0	3_od	1	208	0	0	-1.66	25898	33	LOWERBD	NONBASIC
3	qm	0.33333	1	0	3_qm	1	164	0	0	-1.66	27656	33	LOWERBD	NONBASIC
3	sc	0.33333	1	0	3_sc	1	255	0	0	-1.66	29414	33	LOWERBD	NONBASIC
3	tc	0.33333	1	0	3_tc	1	150	0	0	-1.66	31172	33	LOWERBD	NONBASIC

OPTMODEL solution to network model

- Math Programming Language (GNU Mathprog/AMPL)
- Nodes data sets, immediately into Indexed Sets
- Arcs array and Decision variable Array x[i,j]
- Objective function and Constraints

Data sets…

```
data cadets_loaded;
  infile cadet DLM=',' DSD MISSOVER;
  input OMS $ SEX $ RACE $ prevbr $ BR1 $ BR2 $ BR3 $ BR4 $ BR5 $ acd $ ;
  if _N_ > 1;
  omsnbr = input(OMS,9.);
  ac = input(acd,9.);
  RCE='1';
  if RACE ne '1' then RCE='2';
run;

proc sort data=cadets_loaded;
   by descending omsnbr;

data cadets;
  set cadets_loaded;
  rank=_N_;
run;
```

```
filename goal './ins/goal.txt';
/* this reads in the demand data goals.txt */
data goalst(drop=goal);
  infile goal missover;
  input BR $1-2 goal $ ;
  goalnbr = input(goal,9.);
run;
```

Data sets…expanding goals

```
proc sql; select count(*) as cnt into :all_cnt from cadets; quit;
proc sql; select count(*) into :fem_cnt from cadets where SEX='F'; quit;
proc sql; select avg(omsnbr) into :avg_oms from cadets; quit;
proc sql; select count(*) into :a1_cnt from cadets where ac = 1 ; quit;
proc sql; select count(*) into :a2_cnt from cadets where ac = 2 ; quit;
proc sql; select count(*) into
proc sql; select count(*) into
proc sql; select count(*) into
proc sql; select count(*) into
proc sql; select count(*) into
proc sql; select count(*) into
proc sql; select count(*) into
proc sql; select sum(goalnbr)
```

```
data goals;
  set goalst;
  femgoals=round(goalnbr * &fem_cnt/(&all_cnt-&ca_cnt),1);
  qgoals=0.998*goalnbr*&avg_oms;
  if BR in ('IN' 'AR' 'FA') then femgoals = 0;
  mingoals=round(goalnbr * &min_cnt/&all_cnt,1);
  min2goals=round(goalnbr * &min2_cnt/&all_cnt,1);
  min3goals=round(goalnbr * &min3_cnt/&all_cnt,1);
  min4goals=round(goalnbr * &min4_cnt/&all_cnt,1);
  min5goals=round(goalnbr * &min5_cnt/&all_cnt,1);
  a1goals=round(goalnbr * &a1_cnt/&all_cnt,1);
  a2goals=round(goalnbr * &a2_cnt/&all_cnt,1);
  a3goals=round(goalnbr * &a3_cnt/&all_cnt,1);
  a4goals=round(goalnbr * &a4_cnt/&all_cnt,1);
  qgoals=0.998*goalnbr*&avg_oms;
run;
```

to Indexed Set for Cadet Nodes

```
proc optmodel printlevel=2;
  set C;
  string OMS{C};
  string SEX{C};
  string RACE{C};
  string prevbr{C};
  string BR1{C};
  string BR2{C};
  string BR3{C};
  string BR4{C};
  string BR5{C};
  number omsnbr{C};
  number ac{C};
  string RCE{C};
  number rank{C};
  read data cadets into C=[_N_]OMS SEX RACE prevbr
              BR1 BR2 BR3 BR4 BR5 omsnbr ac RCE rank;
  print OMS SEX RACE prevbr BR1 BR2 BR3 BR4 BR5 omsnbr ac RCE rank;
```

[C]	OMS	SEX	RACE	prevbr	BR1	BR2	BR3	BR4	BR5	omsnbr	ac	RCE	rank
1	97.21103	M	X	IN	IN	EN	FA	MP	SC	97.211	1	M	1
2	96.85913	F	C	AV	AV	AG	FI	MS	MI	96.859	2	C	2
3	96.69295	M	C	AV	AV	MS	AR	MI	IN	96.693	2	C	3
4	96.54559	M	C	AV	AV	IN	EN	FA	AR	96.546	4	C	4
5	96.22152	M	C	IN	IN	MI	EN	MP	AR	96.222	1	C	5
6	95.98078	M	C	EN	EN	MI	IN	AR	MS	95.981	2	C	6
7	95.55315	M	C	MP	MP	EN	MI	AR	FA	95.553	1	C	7
8	95.29215	M	C	IN	IN	AR	EN	MI	OD	95.292	4	C	8
9	94.84156	M	X	EN	EN	FI	MI	SC	AR	94.842	1	M	9
	•												
	•												
	•												

to Indexed Set for Goal Nodes

```
set B;
 string BR{B};
 number goalnbr{B};
 number femgoals{B};
 number mingoals{B};
 number min2goals{B};
 number min3goals{B};
 number min4goals{B};
 number min5goals{B};
 number a1goals{B};
 number a2goals{B};
 number a3goals{B};
 number a4goals{B};
 number qgoals{B};
 read data goals into B=[_N_] BR goalnbr femgoals mingoals
    min4goals min3goals min2goals min5goals a1goals a2goals
    a3goals a4goals qgoals;


 print BR goalnbr femgoals mingoals min4goals min3goals min2goals
    min5goals a1goals a2goals a3goals a4goals qgoals;
```

[B]	BR	goalnbr	femgoals	mingoals	min 2goals	min 3goals	min 4goals	min 5goals	a1goals	a2goals	a3goals	a4goals	qgoals
1	AD	82	15	18	**6**	6	5	1	51	18	7	6	6523.6
2	AG	144	26	32	**11**	11	8	1	90	32	12	10	11456.1
3	AR	128	0	28	**10**	10	7	1	80	28	11	9	10183.2
4	AV	146	26	32	**11**	11	8	1	91	32	13	10	11615.3
5	CM	67	12	15	**5**	5	4	1	42	15	6	5	5330.3
6	EN	189	34	42	**15**	15	11	1	118	42	16	13	15036.2
7	FA	233	0	51	**18**	18	14	2	146	51	20	16	18536.7
8	FI	35	6	8	**3**	3	2	0	22	8	3	2	2784.5
9	IN	282	0	62	**22**	22	16	2	177	62	24	19	22435
10	MI	362	65	80	**28**	28	21	3	227	80	31	25	28799.5
11	MS	0	0	0	**0**	0	0	0	0	0	0	0	0
12	MP	100	18	22	**8**	8	6	1	63	22	9	7	7955.7
13	OD	208	37	46	**16**	16	12	2	130	46	18	14	16547.8
14	QM	164	29	36	**13**	13	10	1	103	36	14	11	13047.3
15	SC	255	46	56	**20**	20	15	2	160	56	22	17	20286.9
16	TC	150	27	33	**12**	12	9	1	94	33	13	10	11933.5

Arcs Indexed Set, connects nodes

```
number arc{B,C};

for{i in B, j in C}
if BR[i]=BR1[j] then arc[i,j] = 6/rank[j];
else if BR[i]=BR2[j] then arc[i,j] = 5/rank[j];
else if BR[i]=BR3[j] then arc[i,j] = 4/rank[j];
else if BR[i]=BR4[j] then arc[i,j] = 3/rank[j];
else if BR[i]=BR5[j] then arc[i,j] = 2/rank[j];
else arc[i,j]=1/rank[j];

print arc;
```

```
var x{B,C} >= 0;
```

arcs

B\C	1	2	3	4	5	6	7
1	1	0.5	0.333333	0.25	0.2	0.166667	0.142857
2	1	2.5	0.333333	0.25	0.2	0.166667	0.142857
3	1	0.5	1.333333	0.5	0.4	0.5	0.428571
4	1	3	2	1.5	0.2	0.166667	0.142857
5	1	0.5	0.333333	0.25	0.2	0.166667	0.142857
6	5	0.5	0.333333	1	0.8	1	0.714286
7	4	0.5	0.333333	0.75	0.2	0.166667	0.285714
8	1	2	0.333333	0.25	0.2	0.166667	0.142857
9	6	0.5	0.666667	1.25	1.2	0.666667	0.142857
10	1	1	1	0.25	1	0.833333	0.571429
11	1	1.5	1.666667	0.25	0.2	0.333333	0.142857
12	3	0.5	0.333333	0.25	0.6	0.166667	0.857143
13	1	0.5	0.333333	0.25	0.2	0.166667	0.142857
14	1	0.5	0.333333	0.25	0.2	0.166667	0.142857
15	2	0.5	0.333333	0.25	0.2	0.166667	0.142857
16	1	0.5	0.333333	0.25	0.2	0.166667	0.142857

...

`setof` function to create subsets

```
set F = setof{i in C: SEX[i]='F'}<i>;

set A1 = setof{i in C: AC[i]=1}<i>;
set A2 = setof{i in C: AC[i]=2}<i>;
set A3 = setof{i in C: AC[i]=3}<i>;
set A4 = setof{i in C: AC[i]=4}<i>;
set R2 = setof{i in C: RACE[i]='2'}<i>;
set R3 = setof{i in C: RACE[i]='3'}<i>;
set R4 = setof{i in C: RACE[i]='4'}<i>;
set R5 = setof{i in C: RACE[i]='5'}<i>;
set M = setof{i in C: RCE[i]='2'}<i>;

set CA = setof{i in B: BR[i] eq 'IN' OR BR[i] eq 'AR' OR BR[i] eq 'FA'}<i>;
set NCA = setof{i in B: BR[i] ne 'IN' AND BR[i] ne 'AR' AND BR[i] ne 'FA'}<i>;

print{i in CA} BR[i];
print{i in NCA} BR[i];
```

```
[CA]
 3      AR
 7      FA
 9      IN

[NCA]
  1     AD
  2     AG
  4     AV
  5     CM
  6     EN
  8     FI
 10     MI
 11     MS
 12     MP
 13     OD
 14     QM
 15     SC
 16     TC
```

Objective, Constraints and decisions…

```
maximize total_score = sum{i in B, j in C}arc[i,j]*x[i,j];

constraint supply{j in C}: sum{i in B}x[i,j]=1;
constraint demand{i in B}: sum{j in C}x[i,j]>=goalnbr[i];
constraint fems_no_ca{i in CA}: sum{j in F}x[i,j]=0;
constraint fcadem{i in B}: sum{j in F}x[i,j]>=femgoals[i];
constraint quality{i in B}: sum{j in C}omsnbr[j]*x[i,j]>=qgoals[i];
solve;
print x;
```

x

B\C	1	2	3	4	5	6
1	0	0	0	0	0	0
2	0	0	0	0	0	0
3	0	0	0	0	0	0
4	0	1	1	1	0	0
5	0	0	0	0	0	0
6	0	0	0	0	0	1
7	0	0	0	0	0	0
8	0	0	0	0	0	0
9	1	0	0	0	1	0
10	0	0	0	0	0	0
11	0	0	0	0	0	0
12	0	0	0	0	0	0
13	0	0	0	0	0	0
14	0	0	0	0	0	0
15	0	0	0	0	0	0
16	0	0	0	0	0	0

…

Indexed Set back to Data set…

```
  create data solt from [B C]={i in B, j in C: x[i,j]>0} BR[i] rank[j] arc[i,j] ac[j] SEX[j] RACE[j] RCE[j]
     prevbr[j] BR1[j] BR2[j] BR3[j] BR4[j] BR5[j] omsnbr[j];
quit;
data sol;
  set solt(rename=(BR=assigned arc=score));
  choice = '6other';
  if assigned = br1 then choice = '1st';
  else if assigned = br2 then choice = '2nd';
  else if assigned = br3 then choice = '3rd';
  else if assigned = br4 then choice = '4th';
  else if assigned = br5 then choice = '5th';
  _random = RANUNI(0);
  rnd = put(_random,10.8);
  rnk = put(rank,6.);
  obs = put(_N_,6.);
  scorec=put(score,10.8);
run;
```

B	C	assigned	rank	score	ac	SEX	RACE	RCE	prevbr	BR1	BR2	BR3	BR4	BR5	omsnbr	choice	_random	rnd	rnk	obs	scorec
9	1	IN	1	6	1	M	2	2	IN	IN	EN	FA	MP	SC	97.211	1st	0.88712	0.887116	1	1037	6
4	2	AV	2	3	2	F	1	1	AV	AV	AG	FI	MS	MI	96.8591	1st	0.74895	0.748954	2	359	3
4	3	AV	3	2	2	M	1	1	AV	AV	MS	AR	MI	IN	96.693	1st	0.14294	0.142943	3	360	2
4	4	AV	4	1.5	4	M	1	1	AV	AV	IN	EN	FA	AR	96.5456	1st	0.04604	0.046043	4	361	1.5
9	5	IN	5	1.2	1	M	1	1	IN	IN	MI	EN	MP	AR	96.2215	1st	0.20021	0.200213	5	1038	1.2
6	6	EN	6	1	2	M	1	1	EN	EN	MI	IN	AR	MS	95.9808	1st	0.46239	0.462389	6	575	1
			*																		
			*																		
			*																		

Analysis of Output

BR	ASSIGNED	PERCENT	GOAL
AD	82	3.222	82
AG	144	5.6582	144
AR	128	5.0295	128
AV	146	5.7367	146
CM	67	2.6326	67
EN	189	7.4263	189
FA	233	9.1552	233
FI	35	1.3752	35
IN	281	11.0413	281
MI	363	14.2633	363
MP	100	3.9293	100
OD	208	8.1729	208
QM	164	6.444	164
SC	255	10.0196	255
TC	150	5.8939	150

choice	COUNT	PERCENT	CUM
1st	1568	61.611	61.611
2nd	646	25.383	86.994
3rd	250	9.823	96.817
4th	41	1.611	98.428
5th	14	0.550	98.978
6other	26	1.022	100.000

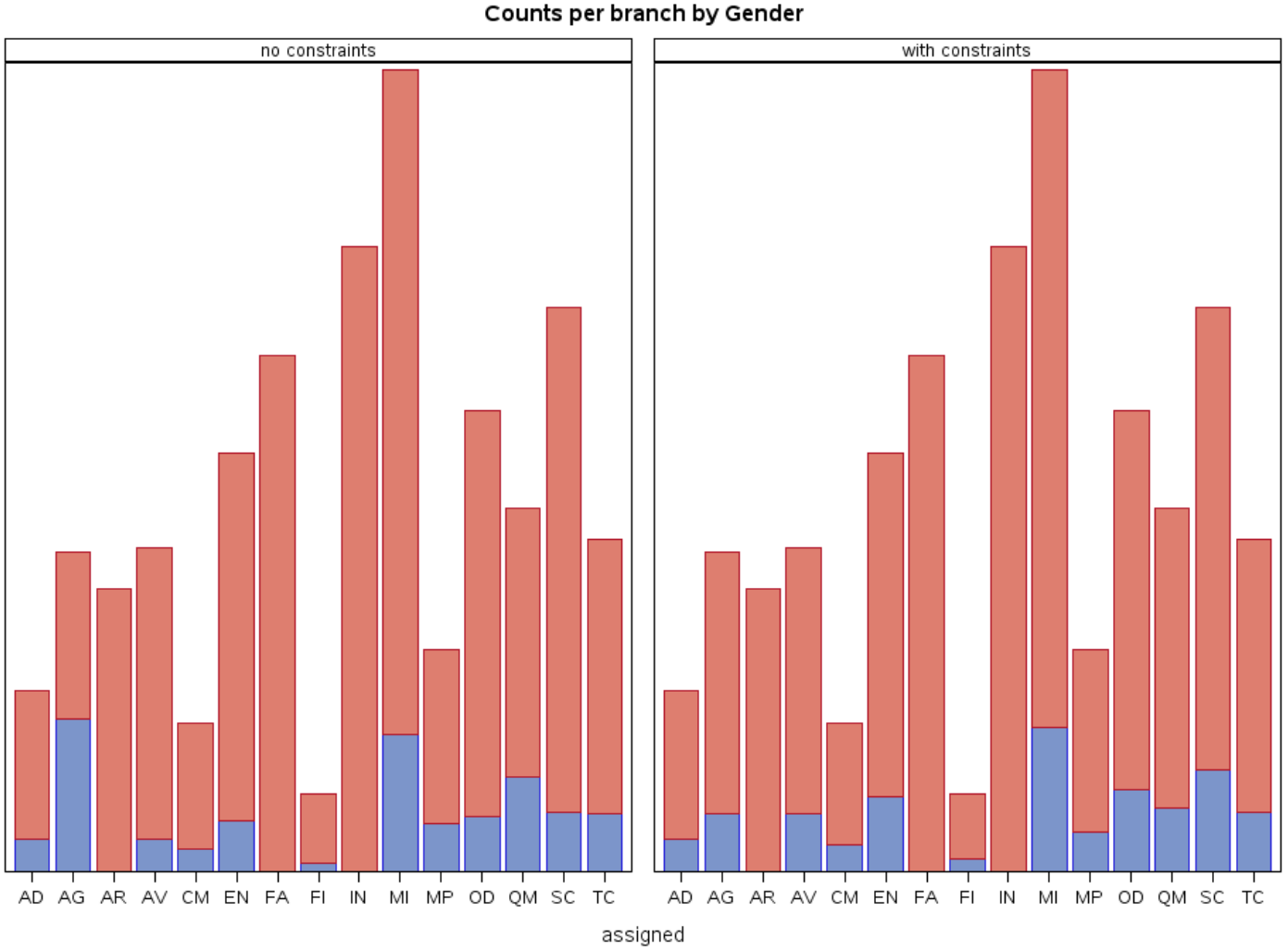
```
maximize total_score = sum{i in B, j in C}arc[i,j]*x[i,j];

constraint supply{j in C}: sum{i in B}x[i,j]=1;
constraint demand{i in B}: sum{j in C}x[i,j]>=goalnbr[i];
constraint fems_no_ca{i in CA}: sum{j in F}x[i,j]=0;

solve;
```

Analysis of Output (gender)

```
constraint fcadem{i in B}: sum{j in F}x[i,j]>=femgoals[i];
```

No constraint					Constrained					Satisfaction	
BBR	ASGN	FEMS	FGOAL	DELTA	BBR	ASGN	FEMS	FGOAL	DELTA		
AD	82	15	15	0	AD	82	15	15	0		
AG	144	69	26	43	AG	144	26	26	0		
AR	128	0	0	0	AR	128	0	0	0		
AV	146	15	26	-11	AV	146	26	26	0		
CM	67	10	12	-2	CM	67	12	12	0	CHOICE	CUMULATIVE
EN	189	23	34	-11	EN	189	34	34	0	1st	60.629
FA	233	0	0	0	FA	233	0	0	0	2nd	85.383
FI	35	4	6	-2	FI	35	6	6	0	3rd	95.678
IN	282	0	0	0	IN	282	0	0	0	4th	97.957
MI	362	62	65	-3	MI	362	65	65	0	5th	98.546
MP	100	22	18	4	MP	100	18	18	0	6other	100
MS	0	0	0	0	MS	0	0	0	0		
OD	208	25	37	-12	OD	208	37	37	0		
QM	164	43	29	14	QM	164	29	29	0		
SC	255	27	46	-19	SC	255	46	46	0		
TC	150	26	27	-1	TC	150	27	27	0		

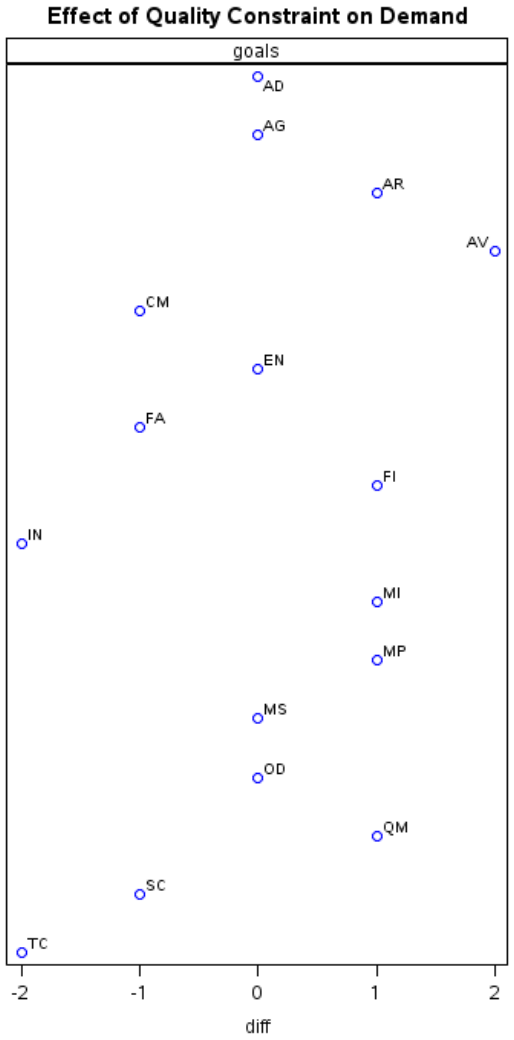
Analysis of Output (gender)



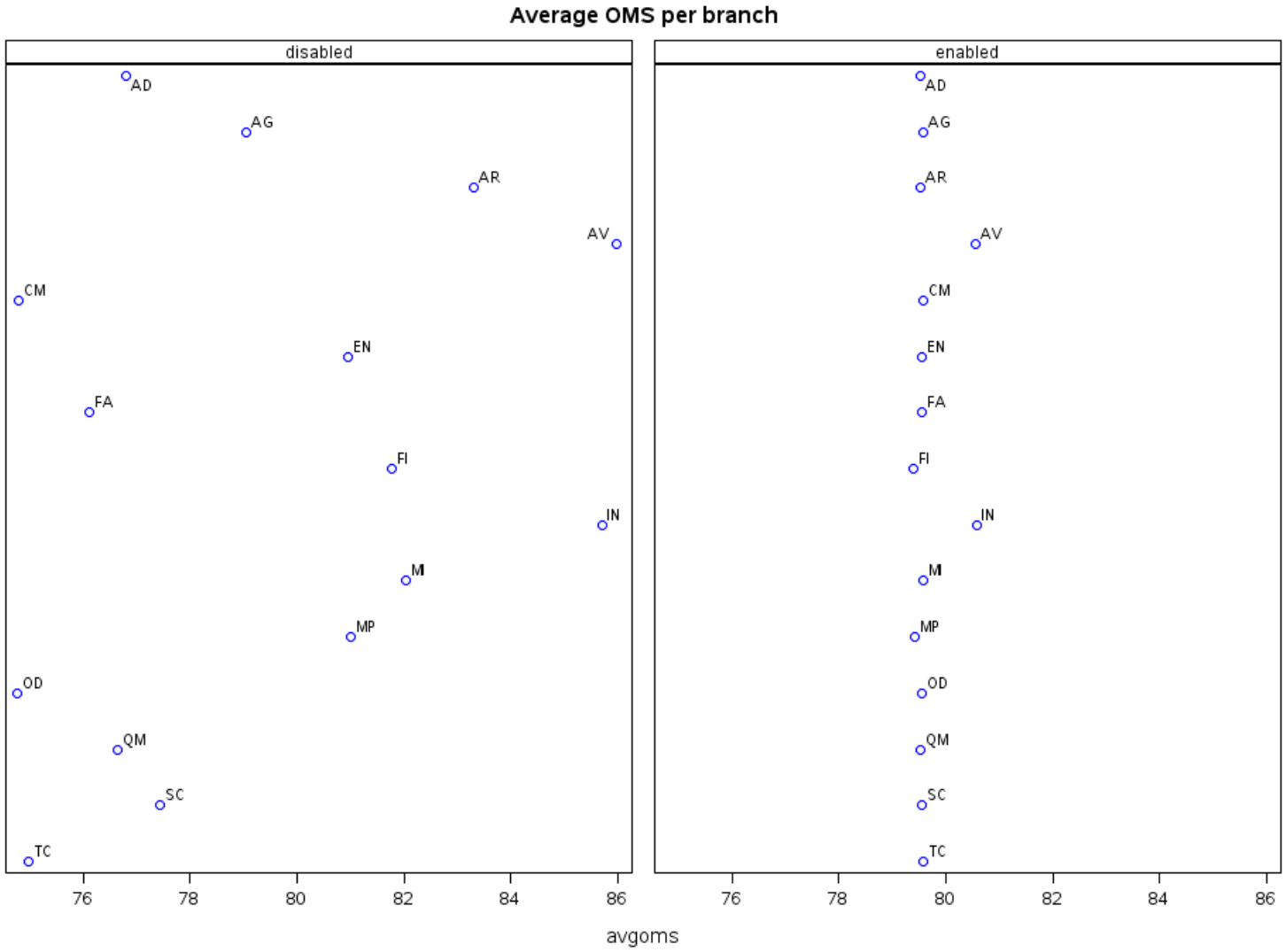
Analysis of Output (quality)

ASSIGNED	AVGOMS	ASSIGNED	AVGOMS
IN	85.69	IN	80.6
AV	85.57	AV	80.43
AR	83.32	OD	79.58
MI	82.03	MI	79.58
FI	81.52	AG	79.57
EN	81.07	AR	79.57
MP	80.85	SC	79.57
AG	79	FI	79.54
SC	77.59	TC	79.53
AD	76.87	MP	79.53
QM	76.85	FA	79.53
FA	76.25	CM	79.52
TC	74.84	QM	79.52
CM	74.71	EN	79.52
OD	74.69	AD	79.48

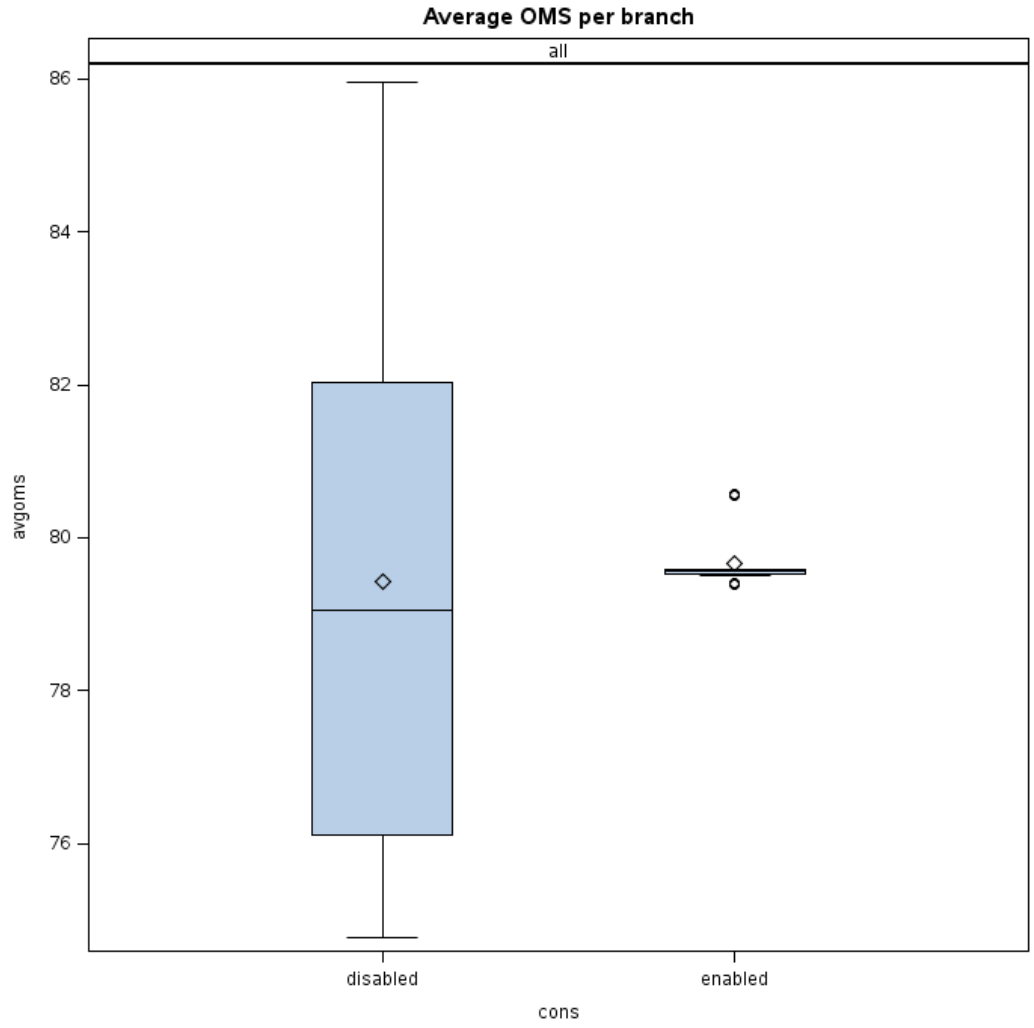
CHOICE	CUMULATIVE
1st	54.892
2nd	78.389
3rd	90.295
4th	93.281
5th	94.931
6other	100



Analysis of Output (quality)



Analysis of Output (quality)



Conclusions

- SAS/OR® supports network optimization,
  - indexed set solutions provided via OPTMODEL
  - Supports Math Programming (Mathprog) Language syntax
  - Even has a data set optimization NETFLOW.
- Base SAS® supports pre and post optimization data analysis.
- Network Optimizations become part of Analysis and Decision process.

2010

APRIL 11 – 14
SEATTLE
WASHINGTON

More about this presentation at http://www.sascommunity.org/

http://www.sascommunity.org/wiki/A_Network_Optimization_Solution_Using_SAS/OR%C2%AE_Tools_for_the_Department_of_the_Army_Branching_Problem

Contact Information

Michael Grierson

mcg@mcg-ct.com

www.mcg-ct.com

Standard Form part 1

```
maximize
  1.00*1_ad + 1.00*1_ag + 1.00*1_ar + 1.00*1_av + 1.00*1_cm + 1.00*1_fi +
  1.00*1_mi + 1.00*1_od + 1.00*1_qm + 1.00*1_tc + 2.00*1_sc + 3.00*1_mp +
  4.00*1_fa + 5.00*1_en + 6.00*1_in + 0.50*2_ad + 0.50*2_ar + 0.50*2_cm +
  0.50*2_en + 0.50*2_fa + 0.50*2_in + 0.50*2_mp + 0.50*2_od + 0.50*2_qm +
  0.50*2_sc + 0.50*2_tc + 1.00*2_mi + 2.00*2_fi + 2.50*2_ag + 3.00*2_av +
  0.33*3_ad + 0.33*3_ag +
  ....
  .0019*2544_ar + .0023*2544_cm + .0003*2545_ag + .0003*2545_av +
  .0003*2545_cm + .0003*2545_fa + .0003*2545_in + .0003*2545_mp +
  .0003*2545_od + .0003*2545_qm + .0003*2545_sc + .0003*2545_tc +
  .0007*2545_mi + .0011*2545_en + .0015*2545_ar + .0019*2545_fi +
  .0023*2545_ad
```

Standard Form part 2

subject too

```
1_ad + 2_ad + 3_ad + 4_ad + 5_ad + 6_ad + ... + 2543_ad + 2544_ad + 2545_ad = 82
1_ag + 2_ag + 3_ag + 4_ag + 5_ag + 6_ag + ... + 2543_ag + 2544_ag + 2545_ag = 144
1_ar + 2_ar + 3_ar + 4_ar + 5_ar + 6_ar + ... + 2543_ar + 2544_ar + 2545_ar = 128
1_av + 2_av + 3_av + 4_av + 5_av + 6_av + ... + 2543_av + 2544_av + 2545_av = 146
1_cm + 2_cm + 3_cm + 4_cm + 5_cm + 6_cm + ... + 2543_cm + 2544_cm + 2545_cm = 67
1_en + 2_en + 3_en + 4_en + 5_en + 6_en + ... + 2543_en + 2544_en + 2545_en = 189
1_fa + 2_fa + 3_fa + 4_fa + 5_fa + 6_fa + ... + 2543_fa + 2544_fa + 2545_fa = 233
1_fi + 2_fi + 3_fi + 4_fi + 5_fi + 6_fi + ... + 2543_fi + 2544_fi + 2545_fi = 35
1_in + 2_in + 3_in + 4_in + 5_in + 6_in + ... + 2543_in + 2544_in + 2545_in = 282
1_mi + 2_mi + 3_mi + 4_mi + 5_mi + 6_mi + ... + 2543_mi + 2544_mi + 2545_mi = 362
1_mp + 2_mp + 3_mp + 4_mp + 5_mp + 6_mp + ... + 2543_mp + 2544_mp + 2545_mp = 100
1_od + 2_od + 3_od + 4_od + 5_od + 6_od + ... + 2543_od + 2544_od + 2545_od = 208
1_qm + 2_qm + 3_qm + 4_qm + 5_qm + 6_qm + ... + 2543_qm + 2544_qm + 2545_qm = 164
1_sc + 2_sc + 3_sc + 4_sc + 5_sc + 6_sc + ... + 2543_sc + 2544_sc + 2545_sc = 255
1_tc + 2_tc + 3_tc + 4_tc + 5_tc + 6_tc + ... + 2543_tc + 2544_tc + 2545_tc = 150
```

Standard Form part 3

subject too

```
1_ad + 1_ag + 1_ar + 1_av + 1_cm + 1_en + 1_fa + 1_fi + 1_in + 1_mi + 1_mp + 1_ms + 1_od
+ 1_qm + 1_sc + 1_tc = 1

2_ad + 2_ag + 2_ar + 2_av + 2_cm + 2_en + 2_fa + 2_fi + 2_in + 2_mi + 2_mp + 2_ms + 2_od
+ 2_qm + 2_sc + 2_tc = 1

3_ad + 3_ag + 3_ar + 3_av + 3_cm + 3_en + 3_fa + 3_fi + 3_in + 3_mi + 3_mp + 3_ms + 3_od
+ 3_qm + 3_sc + 3_tc = 1

...

2545_ad + 2545_ag + 2545_ar + 2545_av + 2545_cm + 2545_en + 2545_fa + 2545_fi + 2545_in
+ 2545_mi + 2545_mp + 2545_ms + 2545_od + 2545_qm + 2545_sc + 2545_tc = 1
```