Changing Methodology

The results of [FM3] speeding up each loop makes it clear that we have to use a different method than dcg through the whole way.

We recall that the vast majority of changes necessary to the problem is through the jobs. The coverage barely changes, but we need to adapt to the jobs.

Last week I suggested 5 algorithms. They all involve first using dcg, then using a combination of jobs-only and cov-only algorithms. *But what if we reversed this?* What if we got the best jobs solution possible through a jobs-only algorithm, and then used the dcg to find if we can dig any deeper in the hole?

The reason this will guaranteed get us the optimal answer is because this is convex optimization. In other words, if we stand at a point where we can't go any further, it is a global maximum. So if I throw in a bunch of good routes, we will inevitably get to the same place as if I threw a bunch of bad routes.

Furthermore, conceptually, if I am doing dcg, I am forced to perform the subproblem on both jobs and coverage. Convexity does not "exempt" us from a slightly worse jobs route set and a mildly better coverage route set.

We do the following:

Mixed DCG: Run jobs-only until you get a final z and final route system. Punch that final route system and coverage system into the dcg and run to completion.

Let's see how we do.

We will measure: Time taken for jobs-only, number of iterations, objectives.
Time taken for dcg, number of iterations, z-objective, x-objective, full objective.

Later, we will run a pure-dcg experiment which will be painful, so we can see how much better the mixed dcg is than pure dcg.

Time is measured in seconds. t/t refers to the ratio of time taken between mixed column generation and jobs-only column generation.

n	m	z_obj	x_obj	full_obj	Time			t/t
		SP = NyVA = mix = dcg	mix = dcg	mix = dcg	mix	dcg	jobs	
10	20	2'262	2'024	4'287	2.747	2.131	0.58	5
11	24	2'658	2'416	5'074	1.218	0.186	0.01	122
12	24	2'717	2'416	5'133	1.239	0.273	0.01	124
13	26	2'987	2'611	5'598	1.199	0.325	0.01	120
14	26	3'035	2'695	5'730	1.203	0.423	0.01	120
15	19	2'539	2'248	4'788	1.106	0.674	0.02	55
16	23	2'566	2'607	5'173	1.417	1.505	0.03	47
17	23	3'012	2'607	5'620	1.452	1.85	0.03	48
18	23	3'014	2'640	5'654	1.583	2.601	0.03	53
19	25	3'036	2'640	5'677	1.666	4.382	0.08	21
20	21	3'028	2'648	5'676	2.359	6.021	0.15	16
21	22	3'257	2'648	5'906	1.815	5.667	0.18	10
22	22	3'364	2'860	6'224	1.812	7.682	0.18	10
23	22	3'399	2'860	6'260	1.747	8.254	0.16	11
24	22	3'445	2'860	6'305	1.901	10.834	0.25	8
25	25	2'845	1'931	4'777	4.064	22.561	0.37	11
26	26	2'856	1'931	4'787	8.529	28.83	0.31	28
27	26	2'867	1'931	4'799	4.812	24.209	0.47	10
28	26	2'933	1'931	4'865	7.233	41.158	0.48	15
29	27	3'003	2'184	5'187	9.449	58.983	0.64	15
30	30	3'282	2'852	6'135	6.955	75.333	0.58	12
31	30	3'284	2'852	6'137	13.59	89.251	0.60	23
32	30	3'292	2'852	6'145	22.04	144.704	0.89	25
33	32	3'406	3'230	6'636	22.04	147.637	0.97	23
34	33	3'649	3'230	6'880	51.98	221.778	1.3	40
35	29	3'314	2'743	6'057	40.87	233.298	1.4	29

For higher cases:

n	m	z_obj			x_obj		full_obj		Time			t/t
		NyV	mix	dcg	mix	dcg	mix	dcg	mix	dcg	jobs	
36	31	3'335	3'335	3'335	3'279	3'279	6'615	6'615	46.21	320.346	1.78	26
37	31	3'746	3'765	3'755	3'279	3'279	7'044	7'034	35.47	317.724	1.78	20
38	32	3'967	3'967	3'967	3'279	3'279	7'246	7'246	38.72	351.572	1.67	23
39	32	3'971	3'971	3'971	3'279	3'279	7'250	7'250	85.36	419.128	2.55	33
40	36	3'784	3'784	3'784	3'088	3'088	6'873	6'873	245.21	808.268	4.36	56
41	39		3'857		3'224		7'081		102.57		6.48	16
42	40		3'966		3'224		7'190		258.49		7.4	35
43	40		3'970		3'224		7'194		330.6		8.05	41
44	40		3'986		3'224		7'210		1752		6.93	253

The magic number is 43 for mixed column generation and 39 for double column generation. The results are entirely in accordance, except for n = 37, when rounding errors happen to compound.