

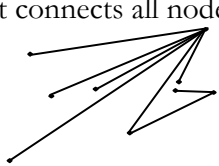
# Min spanning tree in a spreadsheet?

## The limits of spreadsheet modeling

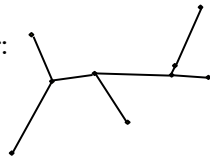
John F. Raffensperger, john.raffensperger@canterbury.ac.nz,  
Dept. of Mgmt, Priv. Bag 4800, Univ. of Canterbury, Christchurch, NZ.

1. What is the minimum spanning tree?  
Shortest tree that connects all nodes.

Very long!



Much shorter:



Telecom, road networks, TSP.

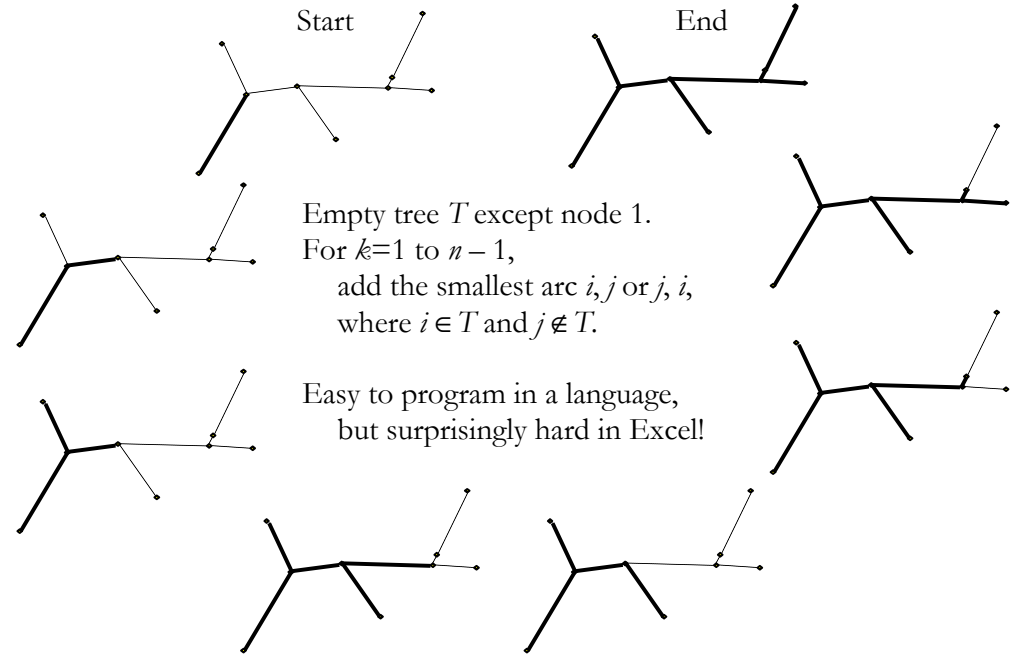
Well-known, widely used, widely taught, easy to solve.

2. MST in a spreadsheet: alternative methods.
3. Held-Karp subgradient optimization.
4. Constructive complexity.

1

Prim's algorithm (1957): find the closest node that doesn't make a cycle.

2



## MST in Excel: options.

3

1. Use VB, e.g., Jensen (2008).
2. Use Solver or What'sBest.
  - standard arc formulation. Break subtours – eeew! Needs VB.
  - Martin's (1991) LP formulation, but  $O(n^3)$  variables.
3. Put Prim's in Excel with formulas.  
Apparently never done before. Why not?

Let's see how:

a spreadsheet row  
for each algorithm step.



Robert Prim William Gates

## Small example. Lots of machinery!

4

I23		=IF(AND(NOT(I14),I15),INDEX(\$B\$4:\$B\$10,MATCH(\$S14,R\$4:R\$10,0),1),"")																							
A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U					
1	Min spanning tree, Dijkstra's algorithm.																	Formula							
2																		Full cost matrix							
3	Costs, from/to	a	b	c	d	e	f	g																	
4	Upper right	a	6.2	7.4	9.5	0.1	9.9	9.0	0.0	6.2	7.4	9.5	0.1	9.9	9.0										
5		b		0.5	1.8	2.8	3.6	0.4	6.2	0.0	0.5	1.8	2.8	3.6	0.4										
6		c			4.9	9.3	6.7	4.5	7.4	0.5	0.0	4.0	0.2	5.7	4.5										
7		d				6.5	1.4	4.2	9.5	0.1	0.0	0.0	0.0	0.0	0.0										
8		e					9.8	8.8	0.1	2.8	9.3	6.5	0.0	9.8	8.8										
9		f						8.2	9.9	3.6	6.7	1.4	9.8	0.0	8.2										
10		g							9.0	0.4	4.5	4.2	8.8	8.2	0.0										
11																Node labels		Cost of next arc							
12	In tree?	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	6.2	7.4	9.5	0.1	9.9	9.0	0.1	
13		2	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	2.8	7.4	6.5	0.0	9.8	8.8	2.8
14		3	1	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15		4	1	1	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
16		5	1	1	1	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
17		6	1	1	1	1	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18		7	1	1	1	1	1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
19																									
20																									
21	From, to	1	.	.	.	.	.	a	.	e	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
22		2	.	e	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
23		3	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
24		4	.	.	b	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
25		5	.	.	.	b	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
26		6	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.

$S12 = \{=MIN(IF(L12:R12<>0,L12:R12))\}$   
 $M17 = \{=MIN(IF(TRANSPOSE(SC17:S117),M54:M510))\}$   
 $D19 = \{=IF(OR(D17,SS17=M17)*SUM(SC17:C17)=SUM(SC18:C18))\}$   
 $S21 = \{=INDEX(SC53:SIS3,1,MATCH(S12,L12:R12,0))\}$

Add node a, then e,  
then b, g, c, d, and f.

Generic method of  
**one row per iteration.**  
Applies to many  
algorithms in Excel.

1. Costs. Save input time with upper  $\frac{1}{2}$  matrix.
2. Costs, full matrix. Easy array formula.
3. **Algorithm.**  
 $n - 1$  steps, so  $n - 1$  rows.  
 Each column is an entering node.  
 Read [4] in the previous row.  
 Complex to break ties.
4. Cost labels on the nodes.  
 Read [3] in the same row.  
 Complex – need a transpose.
5. Output.

50 cities.

Same structure  
as before.

Unwieldy!

Could be  
made smaller  
with more work.

## Held-Karp (1970) subgradient optimization.

Add an extra arc to the MST to make a 1-tree.

For each node  $i$ , if  $\text{degree}[i] \neq 2$ , add a penalty  $\lambda_i$ .  
 Re-draw the 1-tree. Adjust the  $\lambda_i$  penalties again.

Maybe we'll find a tour!

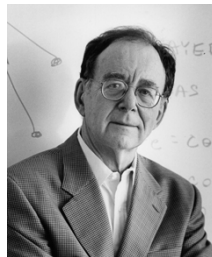
Change the MST spreadsheet to do this.

Method 1: copy the whole MST spreadsheet for each iteration.

May need 100s of iterations. Ugh!

Method 2: use circular references.

More concise – just need one 1-tree block.



Richard Karp

## More concise, from hard work.

$x$ - $y$  coordinates save input time. Node penalties at right.

I simplified blocks for “Cost of next arc” and “Solution”. [Demo]

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	
1	Min spanning 1-tree													1		
2	Formulas are gray													Current	Old	
3					a	b	c	d	e	f	g			stepsize	stepsize	
4				x-coord	17	2	0	25	10	27	59			17.3	17.3	
5			x-coord	y-coord	3	50	63	26	49	0	30			lambda	lambda	Degree
6	Lengths.	a	17	3	0.0	49.3	62.4	24.4	46.5	10.4	49.9			0	0	2
7		b	2	50	49.3	0.0	13.2	33.2	8.1	55.9	60.4			0.0	0.0	2
8		c	0	63	62.4	13.2	0.0	44.7	17.2	68.5	67.6			0.0	0.0	1
9		d	25	26	24.4	33.2	44.7	0.0	27.5	26.1	34.2			0.0	0.0	3
10		e	10	49	46.5	8.1	17.2	27.5	0.0	51.9	52.6			0.0	0.0	2
11		f	27	0	10.4	55.9	68.5	26.1	51.9	0.0	43.9			0.0	0.0	2
12		g	59	30	49.9	60.4	67.6	34.2	52.6	43.9	0.0			0.0	0.0	2
13																
14	Prim's algorithm	1			1	0	0	0	0	0	0	0	0	10.4		
15		2			1	0	0	0	0	1	0	0	0	24.4		
16		3			1	0	0	1	0	1	0	0	0	27.5		
17		4			1	0	0	1	1	1	0	0	0	8.1		
18		5			1	1	0	1	1	1	0	0	0	13.2		
19		6			1	1	1	1	1	1	0	0	0	34.2		
20		7			1	1	1	1	1	1	1	0	0	43.9		
21		8			1	1	1	1	1	1	1	1	1	161.6	= tree length	
22	Solution.			From		b	c	d	e	f	g	g				
23				To		e	b	a	d	a	d	f				

# Definition: constructive complexity

is the number of keystrokes to *write* a spreadsheet, as a function of the input data (Raffensperger & Richard 2008).

- The MST algorithm is **solvable** in  $O(m + n \log n)$  time.
- How much effort to **write the program**? Should be  $O(1)$ .
- How many keystrokes to make the **spreadsheet**? Not clear it's  $O(1)$ . Even if  $O(1)$ , explaining its operation is ugly.

	A	B	C	D	E	F	G	H
1	Minimizing makespan for a single batching machine							Formula
2								
3	Index k			1	2	3	4	5
4	Release date, r(k)			0	3	5	8	40
5	Process time, p(k)	0	30	10	9	8	8	1
6		5						70
7		4					38	50
8		3				35	40	49
9		2			33	40	42	48
10	Job	1		30	40	42	43	41
11	Completion time, f(k)	0	30	33	35	38	41	

A dynamic program in Excel. Requires  $O(n^2)$  keystrokes. Raffensperger & Richard (2005).

# Analysis of the MST spreadsheet

Input takes  $O(n^2)$  time, or  $O(n)$  with Euclidean coordinates. Sunk in any case.

1. An easy transpose formula.
2. Current & new step size. Circular ref M4 to N4.
3. Euclidean distance formula.
4. Lambdas, as a function of step size. Circular refs M7:M12 to N7:N12.
5. Prim's algorithm, 1 row per iteration.
6. Cost of next arc, ugly array with transpose.
7. Primal retrieval, ugly INDEX & MATCH formula.

Nothing depends on  $n$ .  $O(1)$  constructive complexity.

Scaling up requires row & col insertion. Rewrite & debug again!

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	To initialize, set this value to 1, press F9. To run, set to 0, press F9.														
2	Formulas are gray.														
3		a	b	c	d	e	f	g	h	i	j	k	l	m	n
4	x-coord	17	2	47	25	10	27	49	2	17	3	17	3	17	3
5	y-coord	3	50	63	26	49	0	30	2	17	3	17	3	17	3
6	Lengths	a	17	3	0.0	49.3	62.4	24.4	46.5	10.4	49.9	0	0	0	2
7		b	2	50	49.3	0.0	13.2	23.2	8.1	55.9	60.4	0.0	0.0	0	2
8		c	0	63	62.4	13.2	0.0	44.7	17.2	68.5	67.6	49.9	0.0	0	1
9		d	25	26	24.4	33.2	44.7	0.0	27.5	26.1	34.2	0.0	0.0	0	3
10		e	10	49	46.5	8.1	17.2	27.5	0.0	51.9	52.6	0.0	0.0	0	2
11		f	27	0	10.4	55.9	60.4	26.1	51.9	0.0	43.9	0.0	0.0	0	2
12		g	49	30	49.9	60.4	67.6	34.2	52.6	43.9	0.0	0.0	0.0	0	2
13															
14	Prim's algorithm	1	0	0	0	0	0	0	0	0	0	0	0	0	10.4
15		2	1	0	0	0	0	1	0	0	0	0	0	0	24.4
16		3	0	0	0	1	0	0	1	0	0	0	0	0	27.5
17		4	0	0	0	0	1	0	0	1	0	0	0	0	34.2
18		5	1	1	1	1	1	1	1	0	0	0	0	0	43.9
19		6	1	1	1	1	1	1	1	0	0	0	0	0	49.9
20		7	1	1	1	1	1	1	1	1	0	0	0	0	49.9
21		8	1	1	1	1	1	1	1	1	1	0	0	0	161.6 = tree length
22	Solution	From	b	c	d	e	f	g	g						
23		To	e	b	a	d	a	d	f						

# Scaling up a spreadsheet means rewriting it!

For  $n$  fixed, using Excel is like using a procedural program. Just change inputs (e.g. distance).

If  $n$  changes, we must **rewrite the spreadsheet**. Even if the # keystrokes is small, rewriting is work & may add errors.

$$n \leftarrow n + 1$$

**The market has spoken.** MST in Excel may be of **little interest**. Complicated formulas. Procedural languages are easy!

**Conclusion:** algorithms may not belong in spreadsheets.

My lack of creativity? Compare Ragsdale's CPM. Also a graph algorithm, but very easy to extend. Nice!

Cliff Ragsdale



# Maybe you can do better!

Referees improved our spreadsheets. If you can do better, let me know! Improving spreadsheets is like improving algorithms! Lots of work & hard to explain.

## References

Gates, William, image from [www.microsoft.com/presspass/images/gallery/execs/web/billg4\\_web.jpg](http://www.microsoft.com/presspass/images/gallery/execs/web/billg4_web.jpg).  
 Held, Michael, and Karp, Richard M., "The Travelling-Salesman Problem and Minimum Spanning Trees," *Operations Research*, vol. 18, no. 6, Nov-Dec 1970, pp. 1138-1162.  
 Jensen, Paul A., 2008, Excel add-ins for *Ops Res Models and Methods*, accessed 14 Aug 2008, [www.me.utexas.edu/~jensen/ORMM/computation/unit/combin/tree\\_span.html](http://www.me.utexas.edu/~jensen/ORMM/computation/unit/combin/tree_span.html).  
 Karp, Richard M., image from <http://www.coe.berkeley.edu/forefront/spring2004/newsmakers.html>.  
 Martin, R. Kipp, "Using Separation Algorithms to Generate Mixed Integer Model Reformulations," *Ops Res. Letters*, 10 (1991) 119-128.  
 Prim, Robert, image from <http://www.ams.org/featurecolumn/images/january2006/trees9.jpg>.  
 Prim, Robert (1957) "Shortest connection networks and some generalisations," *Bell System Tech. J.*, 36, pp1389-1401.  
 Raffensperger, John F., and Richard, Pascal, "Constructive complexity: a metric for OR spreadsheet model design," forthcoming in IJHOME.  
 Raffensperger, John F. & Pascal Richard (2005), "Implementing Dynamic Programs in Spreadsheets," *INFORMS Transactions on Education*, 5(2), [ite.pubs.informs.org/Vol5No2/RaffenspergerRichard](http://ite.pubs.informs.org/Vol5No2/RaffenspergerRichard).  
 Ragsdale C. T. (2003), "A New Approach to Implementing Project Networks in Spreadsheets," *INFORMS Transactions on Education*, 3(3), <http://ite.pubs.informs.org/Vol3No3/Ragsdale>.  
 Ragsdale, C.T., image from <http://www.bit.vt.edu/faculty/ragsdale.html>.