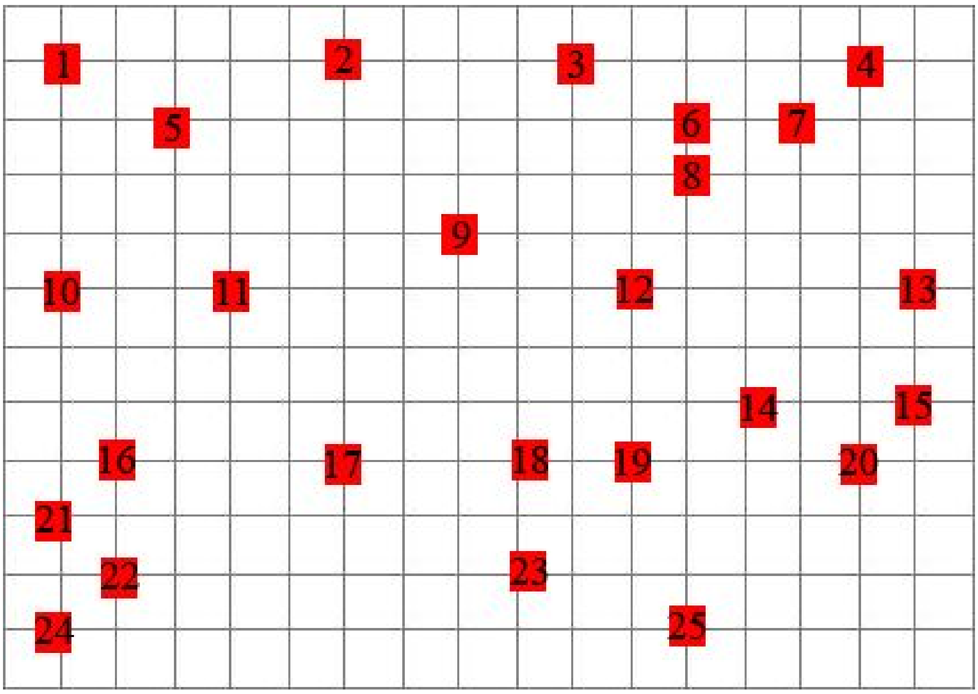
Heuristiek

Chips & Circuits

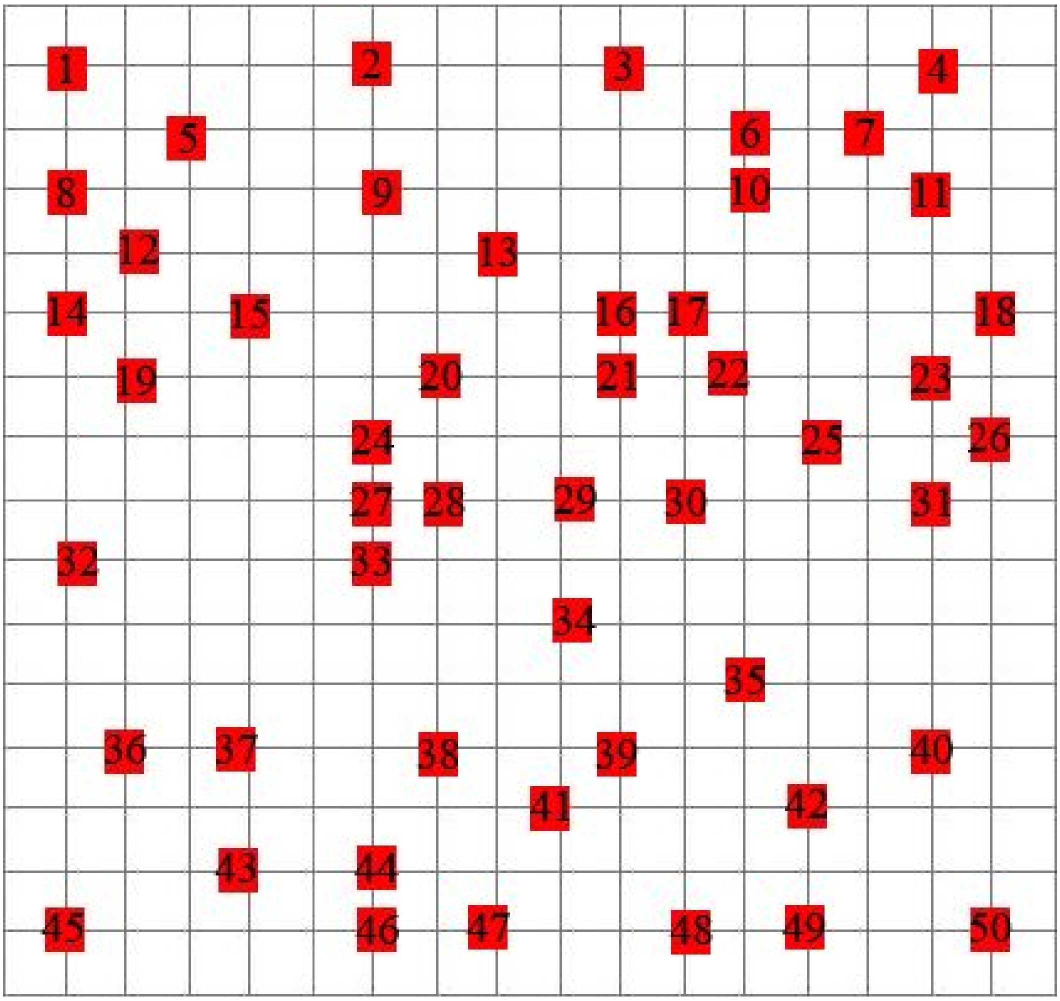
Fish & Circuits
Eelco Alink
Amar Skenderovic
Daniël Staal

Explanation of the problem

This is an integrated circuit built up from five gates that need to be connected. The connections ("nets") follow the grid in a Manhattan style. Luckily enough, the wires don't cross. Sometimes they have to and when they do, the base consists of multiple grid layers. Besides going North, South, East and West, nets can also go up and down, and the distance between levels is identical to the distance between grid points.



Print #1



Print #2

Print #1 and Print #2 are arrangements of gates on a base, and all it takes is to wire the appropriate gates together. There are three net lists (in [txt-format](#)) for each print. Each net list needs to be implemented. Nets can only follow the grid, only one wire per segment, and one step costs 1 unit length. Nets that are aligned among the same grid line are said to be in conflict. If there is one conflict in one arrangement, the circuit cannot be used. Nets can also go up and down to lower and higher layers, also at the cost of 1 per level. The assignment is to implement all nets in all netlists at minimum cost.

A few steps to pave the way towards a program:

1) Build a computer program that holds a data structure for a grid with fixed gates.

2) Expand your program by making a data structure for a net list. Make sure it holds a few nets, and that the program calculates the total wire length.

3) Add 7 more layers. Try to get all the nets in. You can either build up wire-by-wire and step-by,step or with all wires in place, suboptimal and conflictuous, and try to improve. Whatever you do: record data to analyze for your report.

Advanced

For each of the three arrangements, try to determine the relation between the number of wires and the required number of layers.

Questions & Answers (November 2nd, 2014)

1) "only one wire per segment", does this means multiple nets can use a grid intersection?

No, it doesnt. Each grid intersection can support just one net (or "wire"). Still, this might cause problems around the logical gates, so maybe an exception should be made there. We don't know yet, as this is quite a new case.

2) The edges of the prints as shown in the pictures, can we use them to route nets, or should the nets stay clear of the edges?

Yes. You can consider the base to extend a little beyond the picture, and therefore all intersections in the picture can be used.

3) A new layer, should it be somewhere specific, or can we place it where we want?

A new (second, third, fourth) layer provides a means for your nets to go down one level. So, if you add a level to your chip, it is everywhere. But, if a layer (for instance a seventh or eight layer) is unused, it should not be placed of course, to keep manufacturing costs down.

4) Can unconnected gates be removed?

I hope that this issue has been addressed. I expect not to have any unconnected nets left.

5) Can unconnected gates be regarded "null" and may nets run through them?

I hope that this issue too, has been addressed. Otherwise: no!

6) Can loose segments of several logical gates be distributed elsewhere on the base?

No. Even if there are still any of these, I don't want it in this version of Chips and Circuits. Easier versions allowed gates to be moved, and with those degrees of freedom, the case was way too hard.

Exploration of the problem

Restricties:

1. The most optimal configuration needs to be found, where the optimum is the shortest total length of circuits to connect all gates
2. Circuits may not occupy the same space
3. A circuit connecting 2 gates may not cross a third chip
4. Circuits can go under other circuits (on a different layers), where crossing a other circuit means a length of 2 (like left, left) would be a length of 4 (down, left, left, up)

 # print 1
 # Net grid of 18 by 13
 # to calculate the amount of connection on a layer, you need 4(x^2 + x + 1)
(dus voor 18 bij 13 …. Hmmmmm maar mijn formule gaat uit een vierkant)

http://betterexplained.com/articles/navigate-a-grid-using-combinations-and-permutations/

From every gate, there are 6 dimentions (N,S,E,W, up, down)

Every circuit connecting two gates moves within a cube of 18 by 13 by 7 (layers)

The total number of connections for 3 dimensions is (18+13+7)! = 42! = 1.41e51 of 1.41x10^51

This we divide by the dimensions 42!/ (18! * 13! * 7!) = 7, 00e21 of 7,00*10^21

This is for one connection. Every next connection has this value minus the last used possibility… This however only accounts for the the movement one way in every direction. If direction needs to be inverse to avoid collisions, every direction needs movement both ways

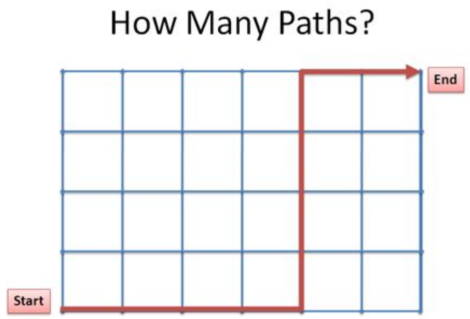
For two directions you need (18+18+13+13+7+7)! = 76!

This makes the total number of possibilities 76! / (18!*18!*13!*13!*7!*7!) = 4,67e52

So…. considering, this should happen for every of the 30 connections that need to be made in the first set of print 1.

What remains is question is how to discount the probabilities where circuits intersect another gate or circuit?

Volgens dit artikel heeft het verbinden van 2 punten in verste hoeken voor print 1 (18x13) dus denk ik:
31!/(18!*13!) = 206253075 = 2.06e8 mogelijkheden

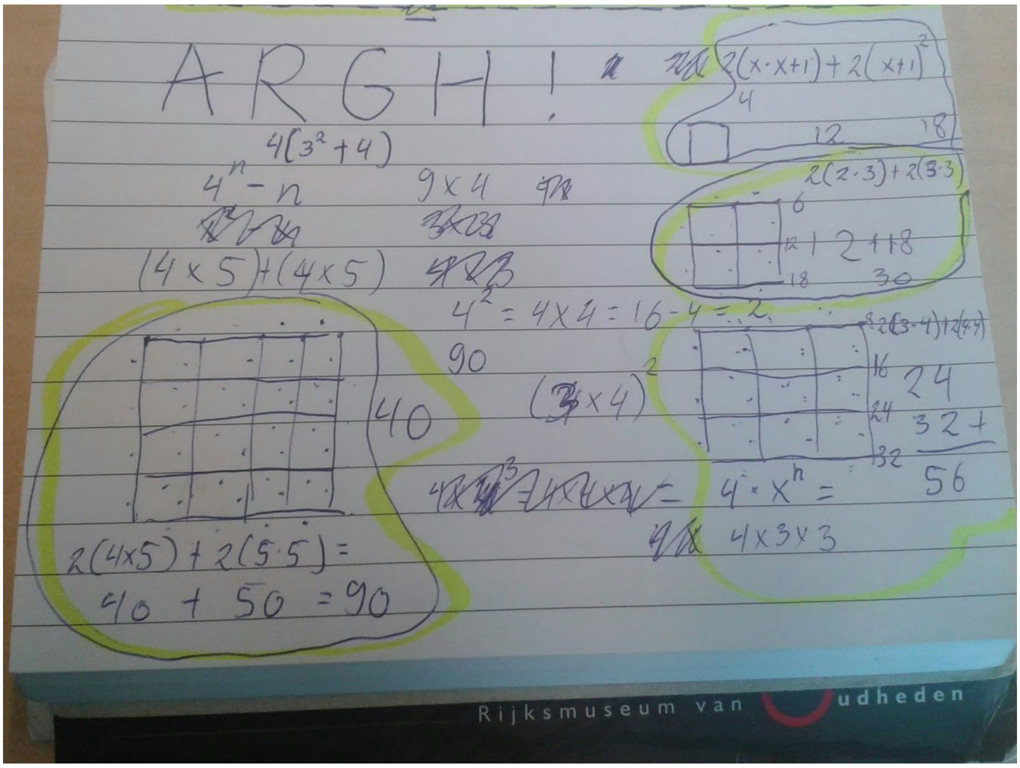


Best veel… En dat zijn er pas 2 en dit is zonder teruggaan denk ik, dus alleen naar rechts en omhoog bijvoorbeeld (misschien is dat wel een goeie aanname)

Met dikte 3 wordt volgens mij alle faculteiten 3x zo groot dus ong. (90!/56!)/40! = 2*10^15

Wellicht wordt het aantal mogelijkheden wel kleiner naarmate we meer punten proberen te verbinden

Dus waar ik eerst uit kwam was 2(x * x+1) (voor plane) + 2((x+1)^2) (voor up/down)



Om te laten zien hoe ik heb zitten rekenen. Dit om te helpen in het maken van een benadering van het maximum aan mogelijkheden.

 # 25 gates

 # lengte 30

netlist_1 = [(23, 4), (5, 7), (1, 0), (15, 21), (3, 5), (7, 13), (3, 23), (23, 8), (22, 13), (15, 17), (20, 10), (15, 8), (13, 18), (19, 2), (22, 11), (10, 4), (11, 24), (3, 15), (2, 20), (3, 4), (20, 19), (16, 9), (19, 5), (3, 0), (15, 5), (6, 14), (7, 9), (9, 13), (22, 16), (10, 7)]

Manual intuitive solution attempt:

- Using one layer rapidly becomes impossible
- Second layer proves not sufficient long long after
- Halted attempt after connection 15 (15, 8), medium (pen, paper, marker) and mental capacity not sufficient to reliably work with more than two layers

 # lengte 40
netlist_2 = [(12, 20), (23, 20), (6, 9), (15, 10), (12, 13), (8, 18), (1, 22), (10, 20), (4, 3), (10, 5), (17, 11), (1, 21), (22, 8), (22, 10), (19, 8), (13, 19), (10, 4), (9, 23), (22, 18), (16, 21), (4, 0), (18, 21), (5, 17), (8, 23), (18, 13), (13, 11), (11, 7), (14, 7), (14, 6), (14, 1), (24, 12), (11, 15), (2, 5), (11, 12), (0, 15), (14, 5), (15, 4), (19, 9), (3, 0), (15, 13)]

 # lengte 50
netlist_3 = [(0, 13), (0, 14), (0, 22), (8, 7), (2, 6), (3, 19), (3, 9), (4, 8), (4, 9), (5, 14), (6, 4), (4, 1), (7, 23), (10, 0), (10, 1), (8, 1), (7, 5), (12, 14), (13, 2), (8, 10), (11, 0), (11, 17), (11, 3), (8, 9), (12, 24), (13, 4), (13, 19), (15, 21), (10, 3), (18, 10), (24, 23), (16, 7), (17, 15), (17, 21), (17, 9), (18, 20), (18, 2), (12, 9), (1, 13), (19, 21), (20, 6), (1, 15), (2, 16), (20, 16), (22, 11), (22, 18), (2, 3), (5, 12), (24, 15), (24, 16)]

 # print 2
 # Net grid of 18 by 16
 # Means a total of 272 circuits per layer
 # 50 gates

#lengte 50
netlist_4 = [(42, 3), (3, 48), (14, 6), (36, 2), (14, 4), (10, 32), (47, 22), (41, 1), (21, 6), (39, 18), (22, 49), (35, 14), (5, 31), (48, 24), (12, 14), (8, 42), (28, 43), (20, 40), (26, 24), (46, 35), (0, 12), (46, 12), (35, 26), (21, 7), (43, 15), (0, 21), (35, 19), (31, 11), (43, 30), (12, 1), (4, 30), (49, 13), (4, 29), (8, 28), (32, 29), (34, 45), (14, 39), (17, 25), (28, 27), (31, 25), (37, 16), (2, 3), (3, 31), (4, 23), (5, 44), (33, 30), (36, 4), (29, 9), (46, 0), (39, 15)]

#lengte 60
netlist_5 = [(34, 21), (48, 47), (38, 16), (0, 16), (28, 40), (24, 8), (36, 37), (26, 8), (8, 27), (39, 48), (44, 34), (22, 30), (43, 44), (47, 5), (19, 30), (31, 41), (0, 10), (12, 32), (3, 33), (45, 18), (0, 21), (23,

43), (44, 42), (18, 11), (24, 23), (41, 13), (26, 1), (16, 1), (20, 29), (31, 4), (7, 28), (28, 45), (0, 12), (44, 29), (34, 5), (2, 17), (9, 5), (30, 9), (36, 29), (18, 27), (32, 11), (40, 10), (4, 40), (35, 6), (17, 3), (10, 19), (25, 24), (20, 47), (12, 25), (4, 15), (19, 33), (33, 36), (1, 3), (13, 49), (25, 49), (15, 42), (33, 4), (27, 22), (4, 8), (12, 24)]

#lengte 70
netlist_6 = [(16, 10), (25, 17), (1, 11), (32, 2), (1, 20), (12, 36), (34, 19), (11, 10), (11, 45), (21, 42), (36, 20), (15, 22), (3, 21), (48, 2), (32, 25), (38, 49), (24, 29), (14, 16), (0, 3), (30, 7), (3, 10), (16, 8), (46, 0), (26, 41), (34, 2), (1, 13), (25, 6), (49, 28), (27, 47), (3, 14), (40, 47), (14, 43), (14, 46), (27, 38), (14, 34), (26, 39), (47, 44), (46, 29), (12, 9), (49, 12), (38, 7), (30, 32), (30, 40), (13, 45), (5, 41), (29, 37), (45, 38), (44, 34), (44, 28), (22, 44), (43, 31), (48, 34), (6, 33), (33, 7), (1, 37), (5, 17), (37, 2), (39, 38), (27, 36), (18, 42), (17, 35), (12, 5), (37, 40), (5, 39), (37, 43), (8, 4), (39, 3), (33, 31), (21, 33), (0, 39)]

Possible helpful literature:
http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=1672647
http://webcache.googleusercontent.com/search?q=cache:http://users.eecs.northwestern.edu/~haizhou/357/lec6.pdf&gws_rd=cr&ei=FOMMV6GFE4W5swGshqToAg
https://en.wikipedia.org/wiki/Shortest_path_problem
https://en.wikipedia.org/wiki/Travelling_salesman_problem#Heuristic_and_approximation_algorithms
http://theory.stanford.edu/~amitp/GameProgramming/AStarComparison.html
http://theory.stanford.edu/~amitp/GameProgramming/Heuristics.html
https://www.researchgate.net/publication/259343167_Sizing_Analog_Integrated_Circuits_by_Current-Branches-Bias_Assignments_with_Heuristics