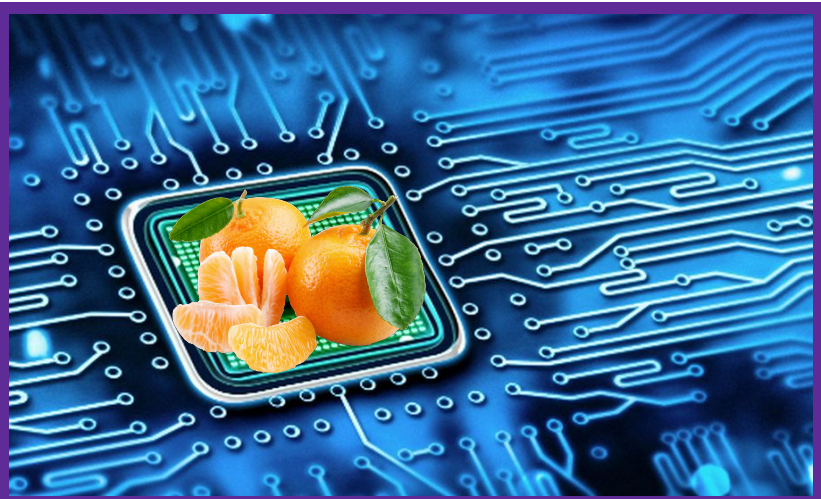
Case: Chips & Circuits

De Mandarijntjes ©



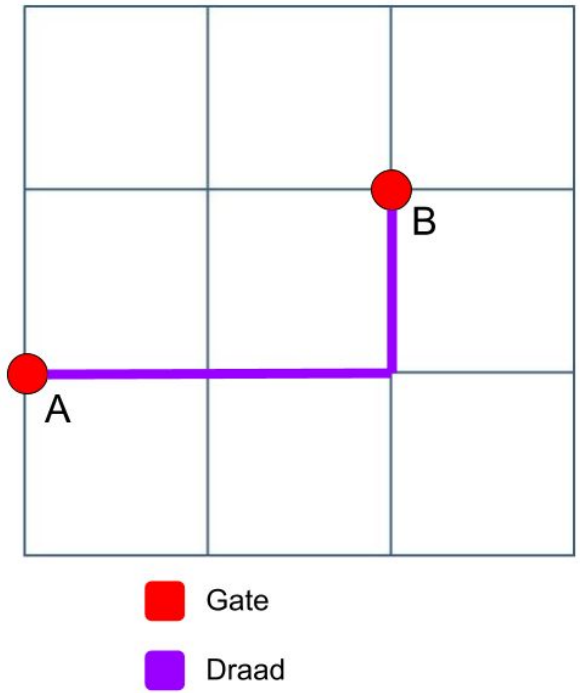
Izhar Hamer
Tom Kamstra
Julia Linde

Introductie

Chip

- Grid
  - Gates
  - Draadjes

- Netlist
- Gate-coördinaten



Netlist

A - B

Probleem: “optimization with constraints”

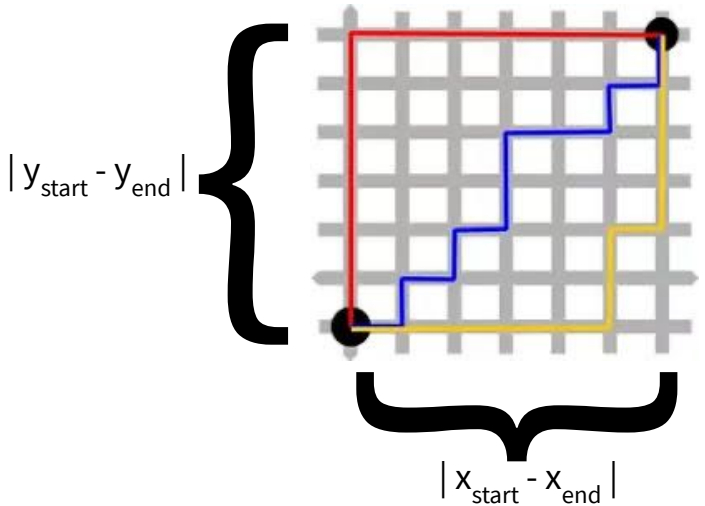
- Doelfunctie = som van alle draadlengtes
  - Minimaliseren
- Constraints:
  - “Manhattan distance”
  - Draden mogen elkaar of andere gates niet raken
  - Maximum van 7 lagen plus onderste laag
  - Draden moeten binnen chip blijven

Onderzoek over case

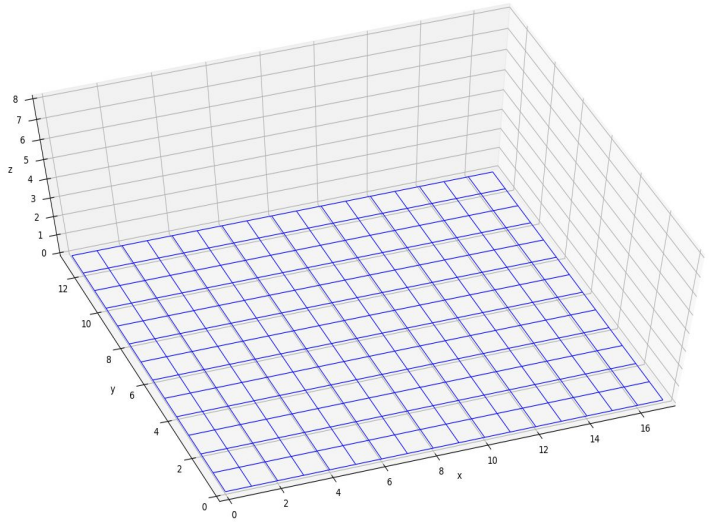
- Lower bound van de doelfunctie:
  - De “Manhattan distance” van iedere connectie tussen gates in de netlist berekenen en optellen:

$$A_i = | x_{start} - x_{end} | + | y_{start} - y_{end} |$$

  - Lower bound: wirelength 291 (voor de netlist met 25 gates en 30 connecties)

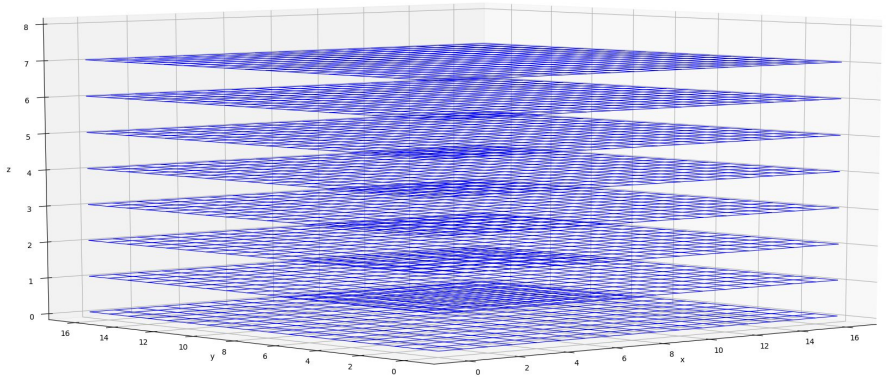


State space



Aantal mogelijkheden = $(L * B * H^{L * B * H}) / 2$

- L = lengte van grid: hoogste y-coördinaat + 1
- B = breedte van grid: hoogste x-coördinaat + 1
- H = hoogte van grid: 7 lagen + onderste laag
- State space: $1.77 * 10^{5741}$ opties (voor de netlist met 25 gates en 30 connecties)
- Geen constraints



Methodes

- Linespacer
  - Linespacer gebaseerd op “Manhattan distance”
  - Linespacer gebaseerd op afstand in x-richting
  - Linespacer gebaseerd op afstand in y-richting
- A*-algoritme
- Breadth first search

Linespacer variaties

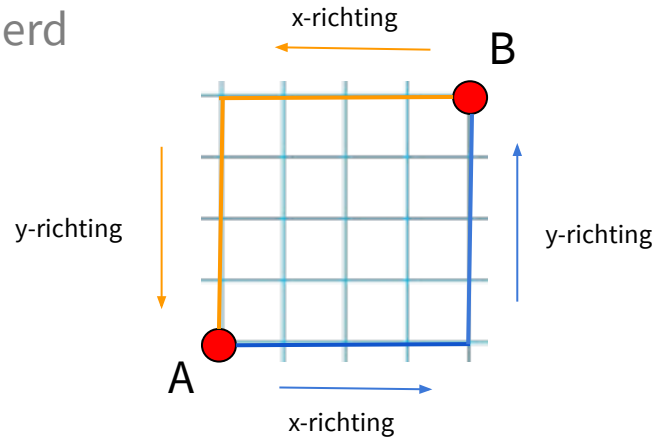
- Linespacer x-richting: netlist gesorteerd op afstand in x-richting
- Linespacer y-richting: netlist gesorteerd op afstand in y-richting
- Linespacer: netlist gesorteerd op totale afstand (“Manhattan distance”)

Linespacer

1. Sorteer netlist gebaseerd op...
2. Verbind gates in vastgestelde volgorde
3. Strip and route
    a. Check of draadje weg kan van start gate
        i. Zo niet: verwijder een draadje en voeg deze later weer toe
4. Laat draadjes laag omhoog gaan als ze andere draden/gates tegenkomen
5. Blijf op hoogst bereikte laag tot draad boven eind gate is
6. Strip and route
    a. Check of draadje weg kan van start gate
        i. Zo niet: verwijder een draadje en voeg deze later weer toe
7. Als draad vastloopt:
    a. Verwijder draden rondom punt waar draad vastloopt en voeg later weer toe
    b. Verwijder aantal langste draden en probeer beter te leggen

Heuristieken van Linespacer

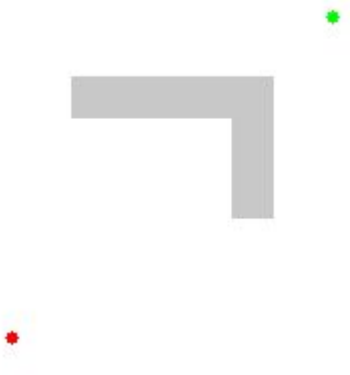
- Gesorteerde netlist
- Beweeg eerst in x-richting, daarna in y-richting
- Verwissel start en eind gate als draad is verwijderd
- Draad is vastgelopen als lengte langer is dan 100
- Draden met langste lengte worden als eerst verwijderd

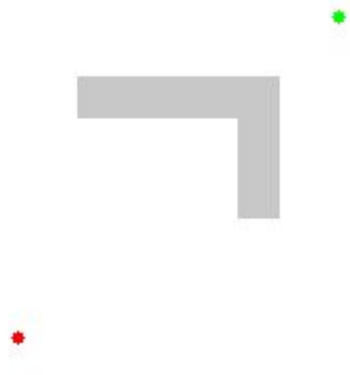


A*-algoritme

- Vergelijkbaar met Dijkstra's algoritme en BFS
  - Verbetering door middel van een heuristiek
- Beslist op basis van F-waarde:
  - F = G + H
    - G = kortste afstand van het begin tot de huidige locatie
    - H = geschatte afstand van de huidige locatie tot het einde
  - Berekent F voor elke buurman
  - Sorteert de mogelijke stappen in "priority queue"

Dijkstra vs. A*





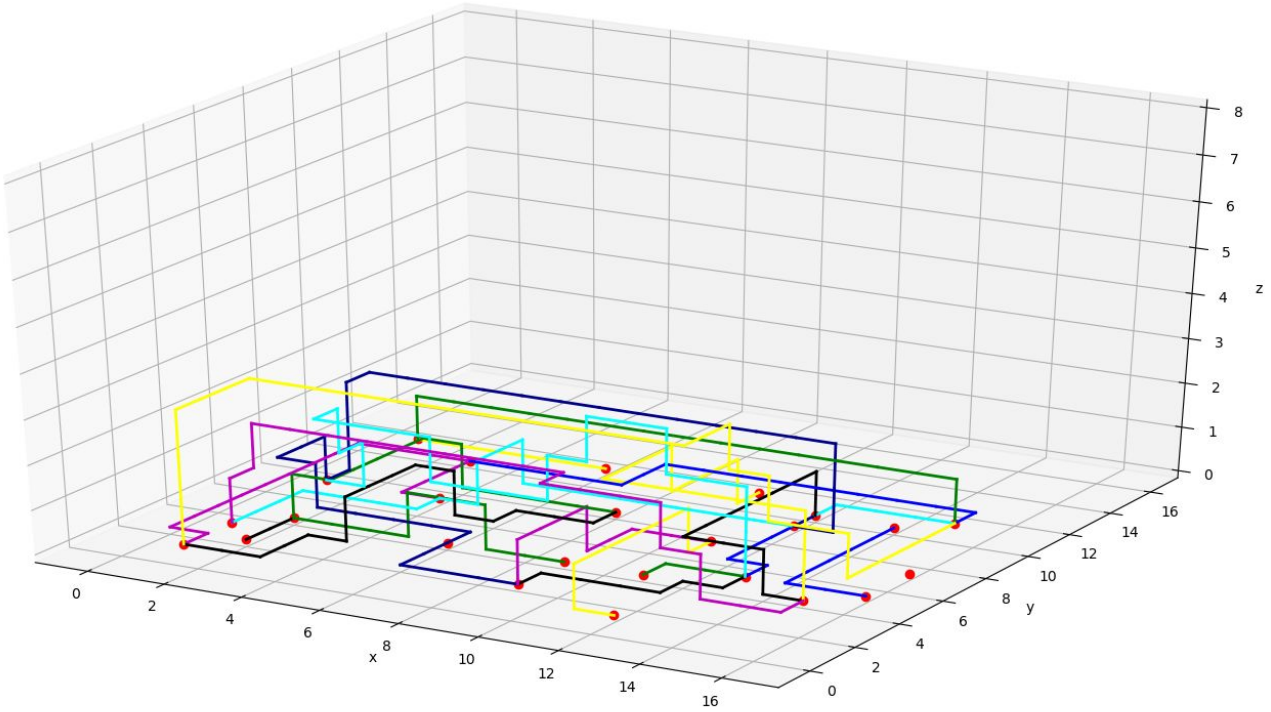
Breadth First Search

- Simpele vorm van A* search
- Doet er te lang over vanwege olievlek methode
- Pruning:
  - Niet alle oplossingen berekenen voor hogere layers als er al een oplossing is op de onderste
- Nosrati, Karimi, & Hasanvand, 2012

Nosrati, M., Karimi, R., & Hasanvand, H. A. (2012). Investigation of the * (Star) Search Algorithms: Characteristics, Methods and Approaches . *World Applied Programming*, *2*(4), 251–256. Geraadpleegd van http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.685.6608&rep=rep1&type=pdf

Heuristiek 1: Kortste draadjes eerst

- Sorteer de netlist gebaseerd op minimale afstand tussen gates



Heuristiek 2: Onderste lagen vermijden

- Verschillende methoden:
  - Draadjes zo hoog mogelijk laten gaan
  - Draadjes naar verschillende layers omhoog niet alleen de bovenste
  - Draadjes alleen omhoog als de geschatte afstand groter is dan 5
  - Onderste lagen duurder maken om overheen te gaan, exclusief stappen over de z-as

Heuristiek 3: Gates vermijden

- Verschillende methoden
  - Directe buren van gates vermijden
  - Ook diagonale buren vermijden
  - Ook alle coördinaten direct boven gates vermijden

Conflict first algoritme

- Wanneer geen directe oplossing gevonden wordt
  - Pak alle draadjes die niet gelegd konden worden
  - Run het programma opnieuw met deze connecties vooraan in de gesorteerde netlist
  - Blijf dit doen totdat er een oplossing gevonden is

Hillclimber

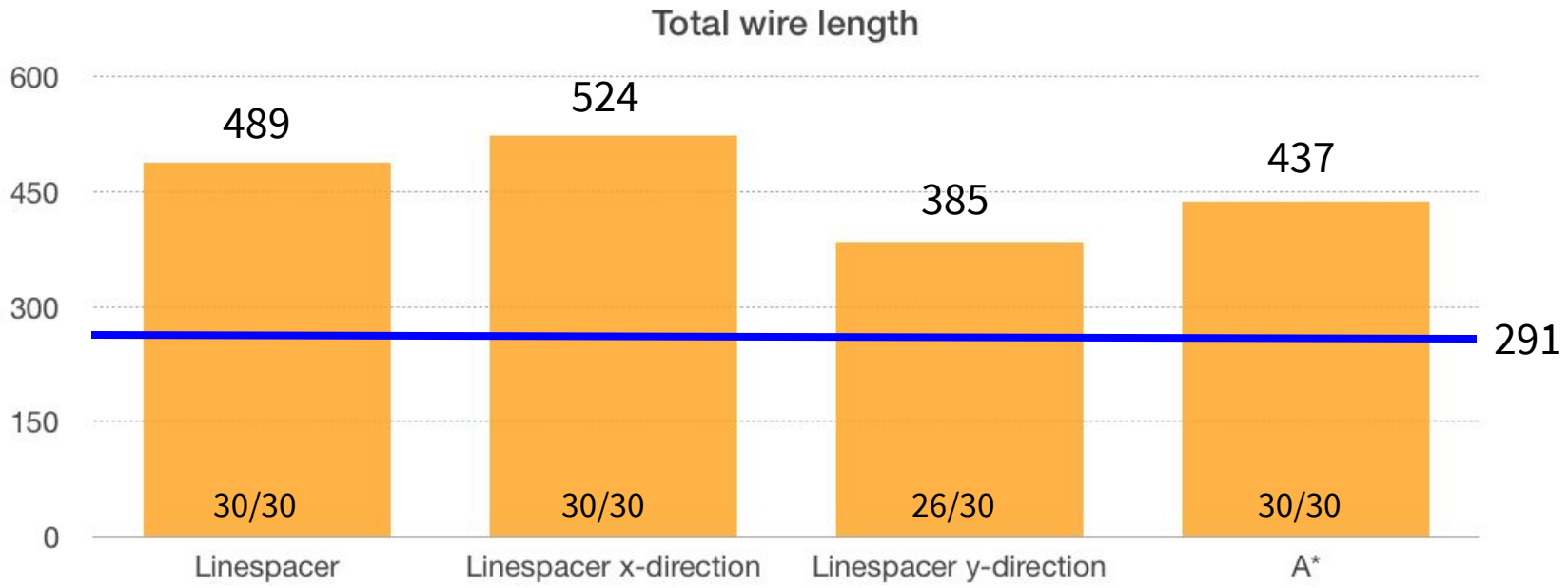
- Wanneer geldige oplossing gevonden is
  - Probeer alle draadjes 1 voor 1 opnieuw te trekken terwijl de rest blijft liggen
  - Doe nu de Astar search zonder de extra heuristieken
  - Als een draadje nu korter wordt, vervang deze voor de oude

Resultaten

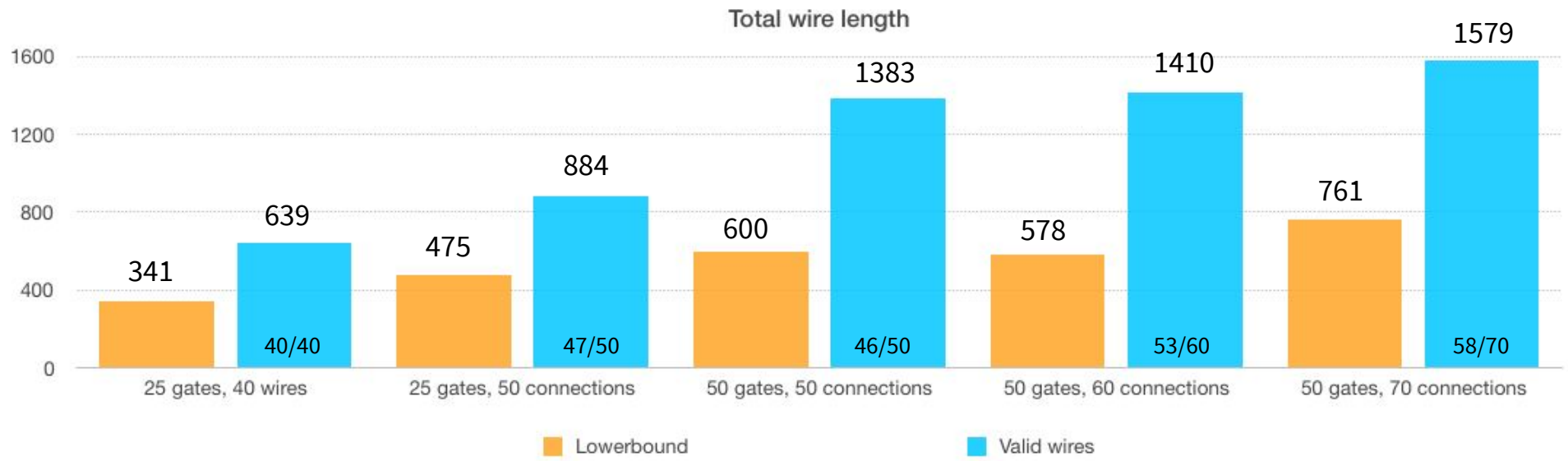
= lower bound

= totale draadlengte per algoritme

Gebaseerd op grid met 25 gates, 30 draden.

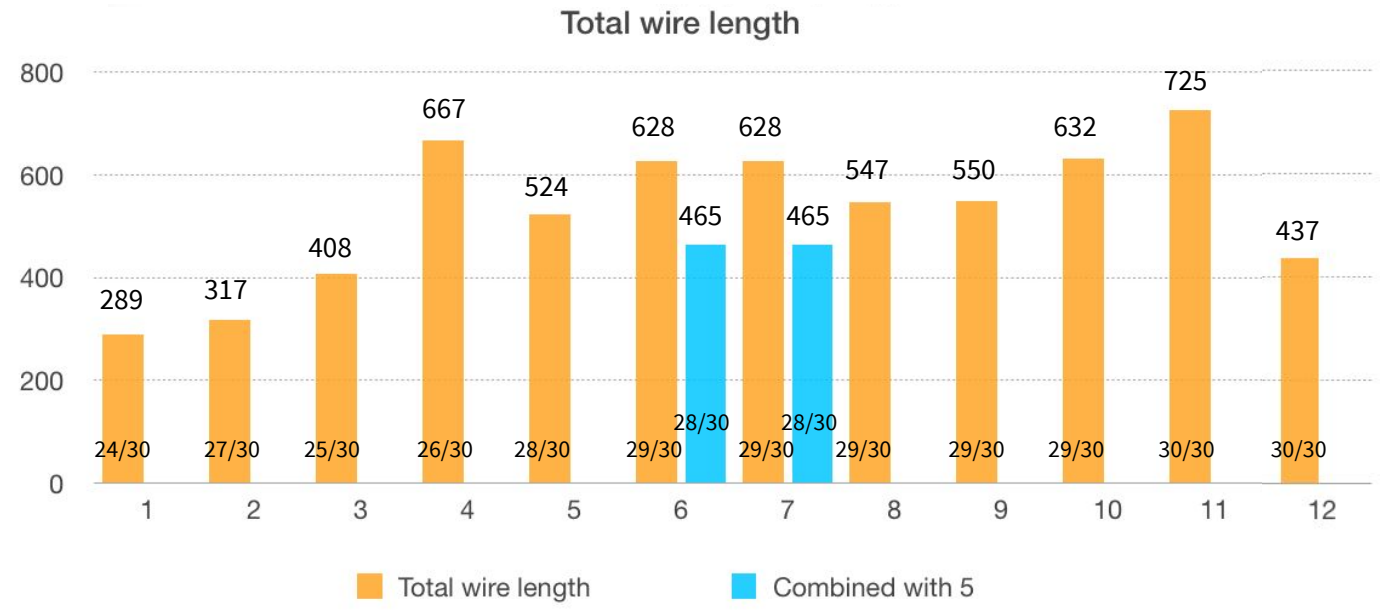


Resultaten A*



Resultaten A* extra toepassingen

Gebaseerd op grid met 25 gates, 30 draden.



1. Random netlist
2. Kortste draden eerst
3. Draden omhoog laten gaan vanaf start gate
4. Draden opzij en dan omhoog als direct omhoog niet kan
5. Draden naar verschillende lagen omhoog in het begin
6. Draden alleen omhoog laten gaan als draad lang genoeg is
7. Onderste lagen duurder maken
8. Directe omgeving van andere gates vermijden
9. Gates op diagonale afstand ook vermijden
10. Z-waarden boven andere gates ook vermijden
11. Conflict first
12. Hillclimber

Exploratie

- Draden direct naar boven sturen geeft meer oplossingen, niet per se betere
  - Vermijd drukte
- Gates met veel draadjes eromheen geven meer problemen
- Problemen met grote gate-dichtheid
- Netlist sorteren op “Manhattan distance” geeft oplossing met kortere totale afstand dan sorteren op x-richting of y-richting

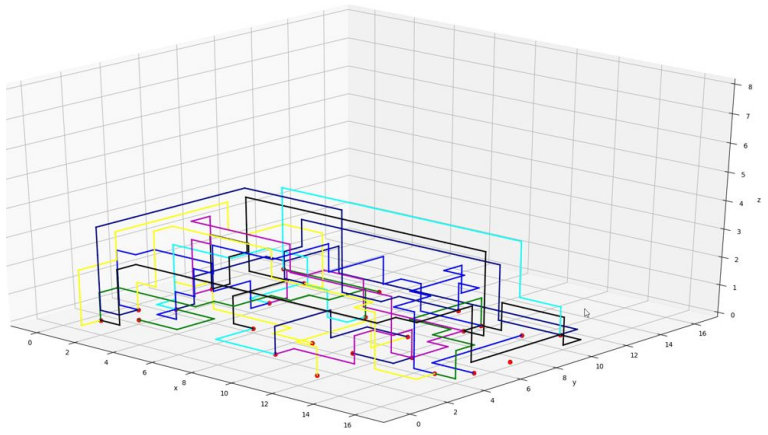
Conclusie

- 25 gates, 30 draden: A* geeft beste oplossing
  - Kortste totale draadlengte
- Andere problemen: geen vergelijking met Linespacer
- Ondanks heuristieken is een constructief algoritme niet genoeg
  - A* zonder heuristieken geeft geen geldige oplossing
  - A* met heuristieken geeft geldige oplossing, maar relatief lange draadlengte
- Iteratief algoritme nodig voor een geldige én relatief goede oplossing
  - Hillclimber

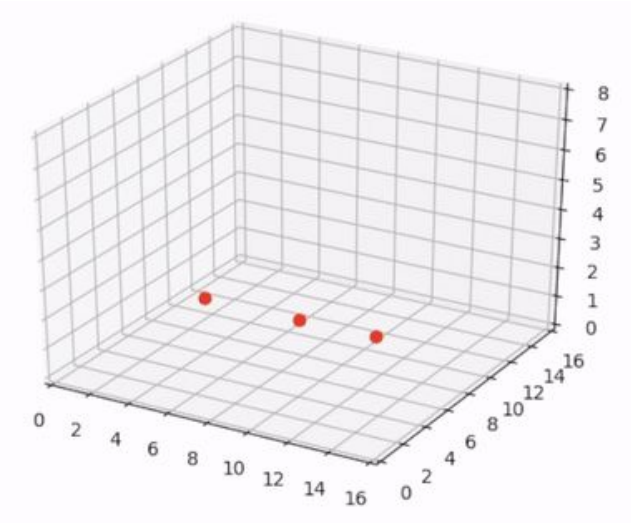
Discussie

- Linespacer: stopt zodra oplossing is gevonden, zoekt niet naar beste oplossing
- Linespacer: geeft laatste oplossing als script te lang loopt, niet beste
- A* houdt geen rekening met volgende draden
  - Beste oplossing op korte termijn

Vragen?

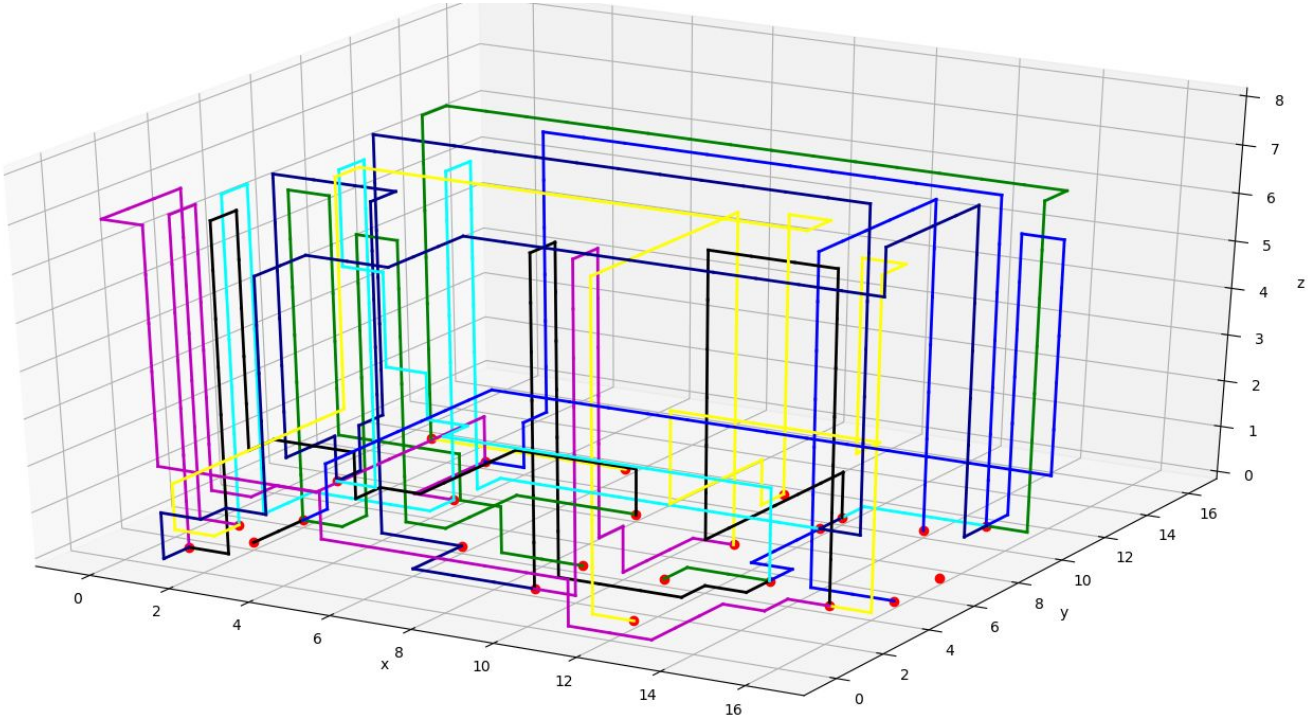


Astar

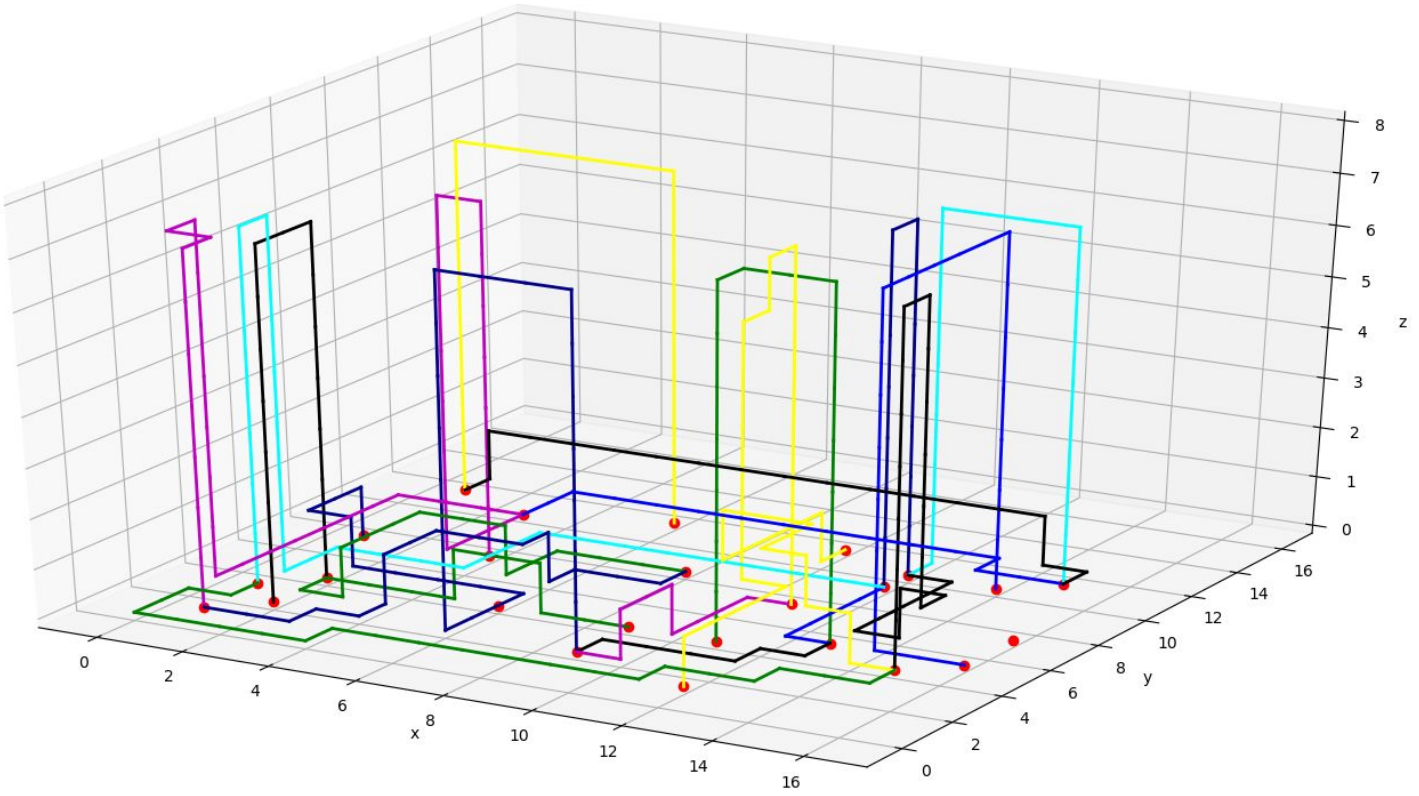


Linespacer

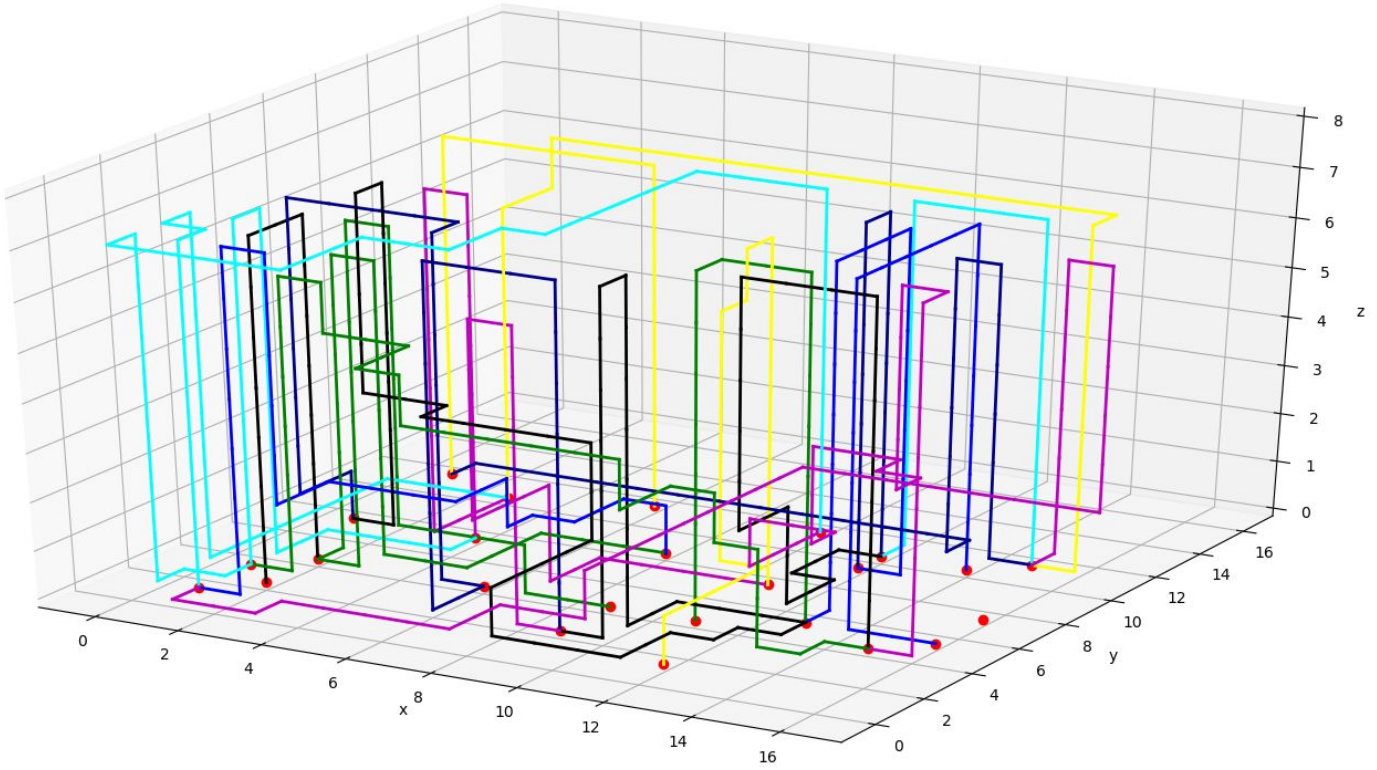
Onderste lagen duurder maken



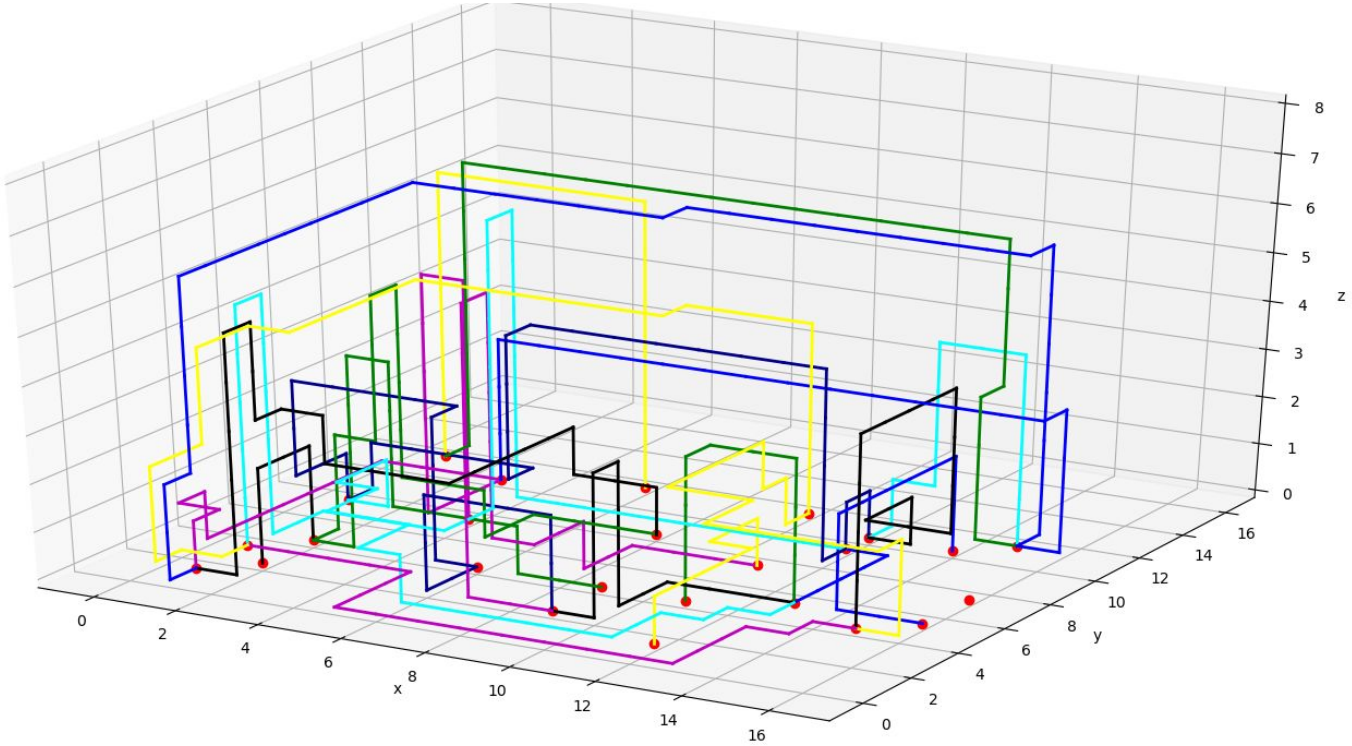
Draadjes zo hoog mogelijk forceren



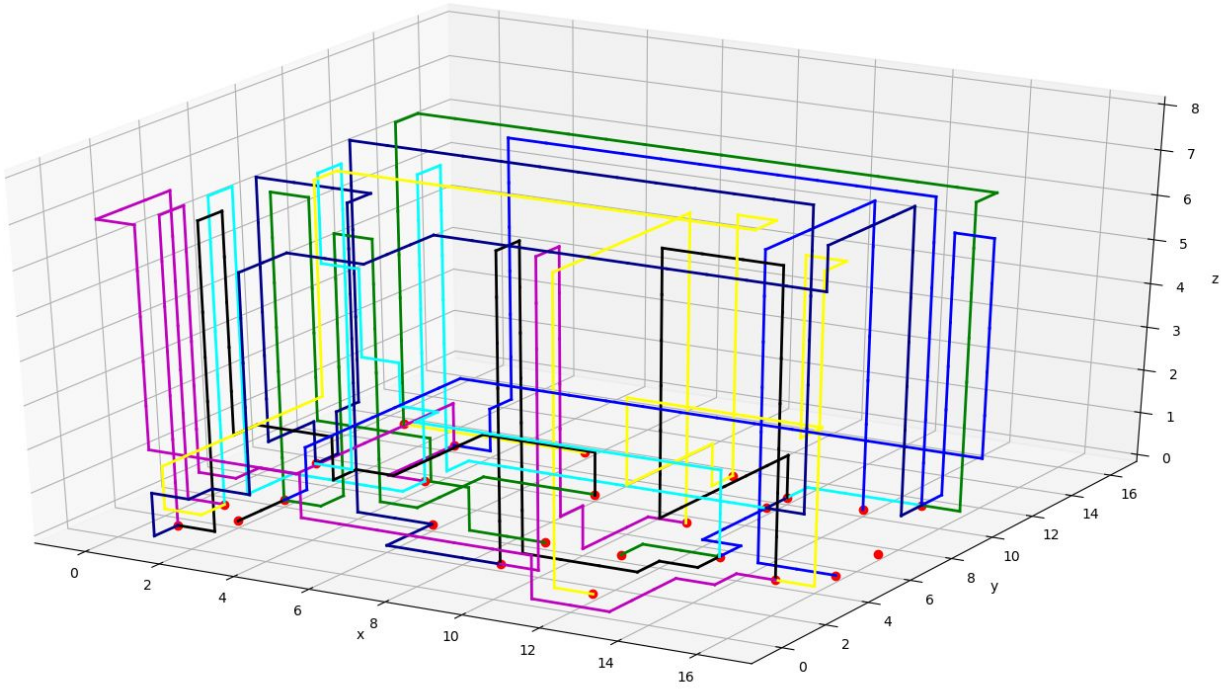
Als direct omhoog niet kan eerst opzij



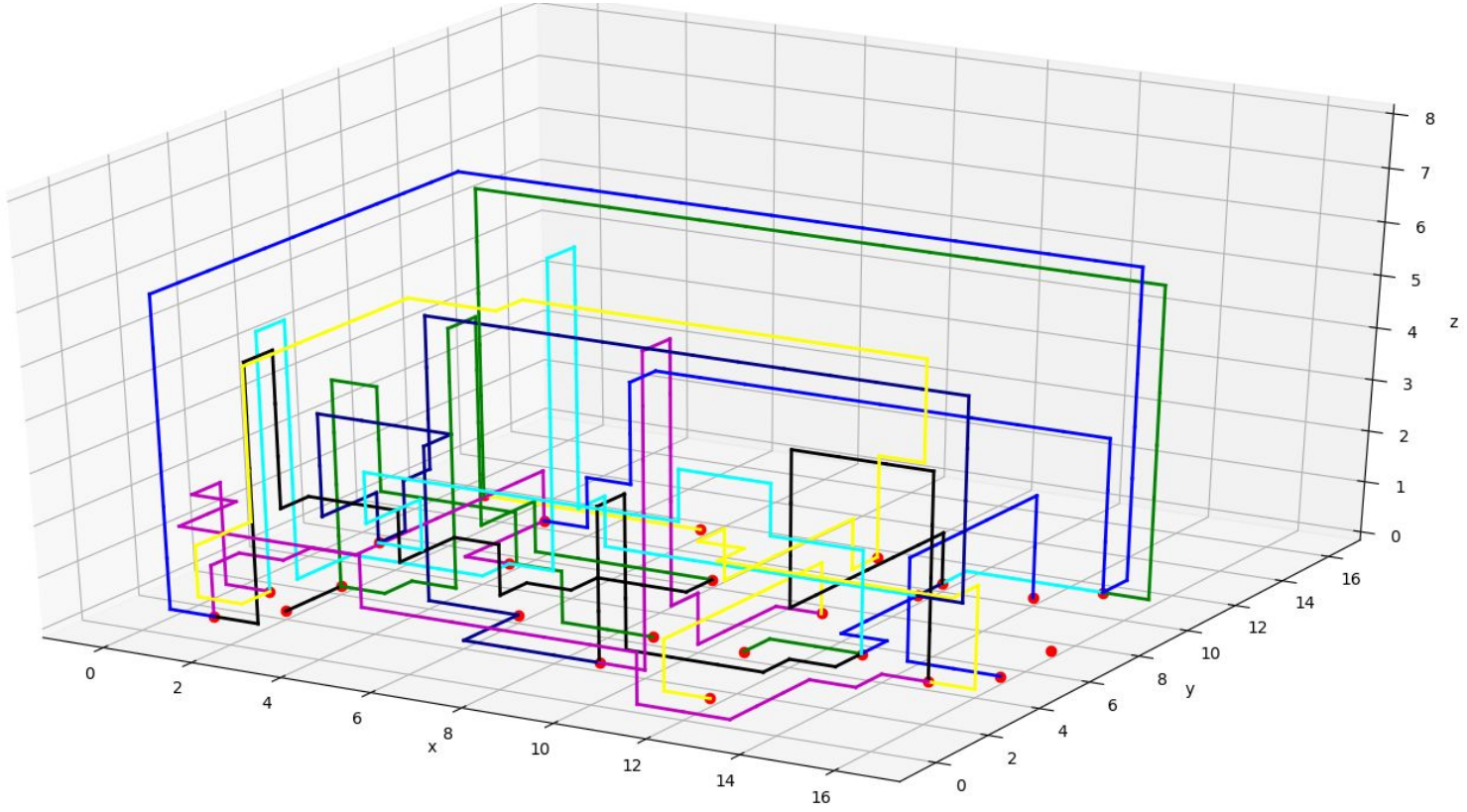
Draadjes naar lagen met verschillende hoogtes



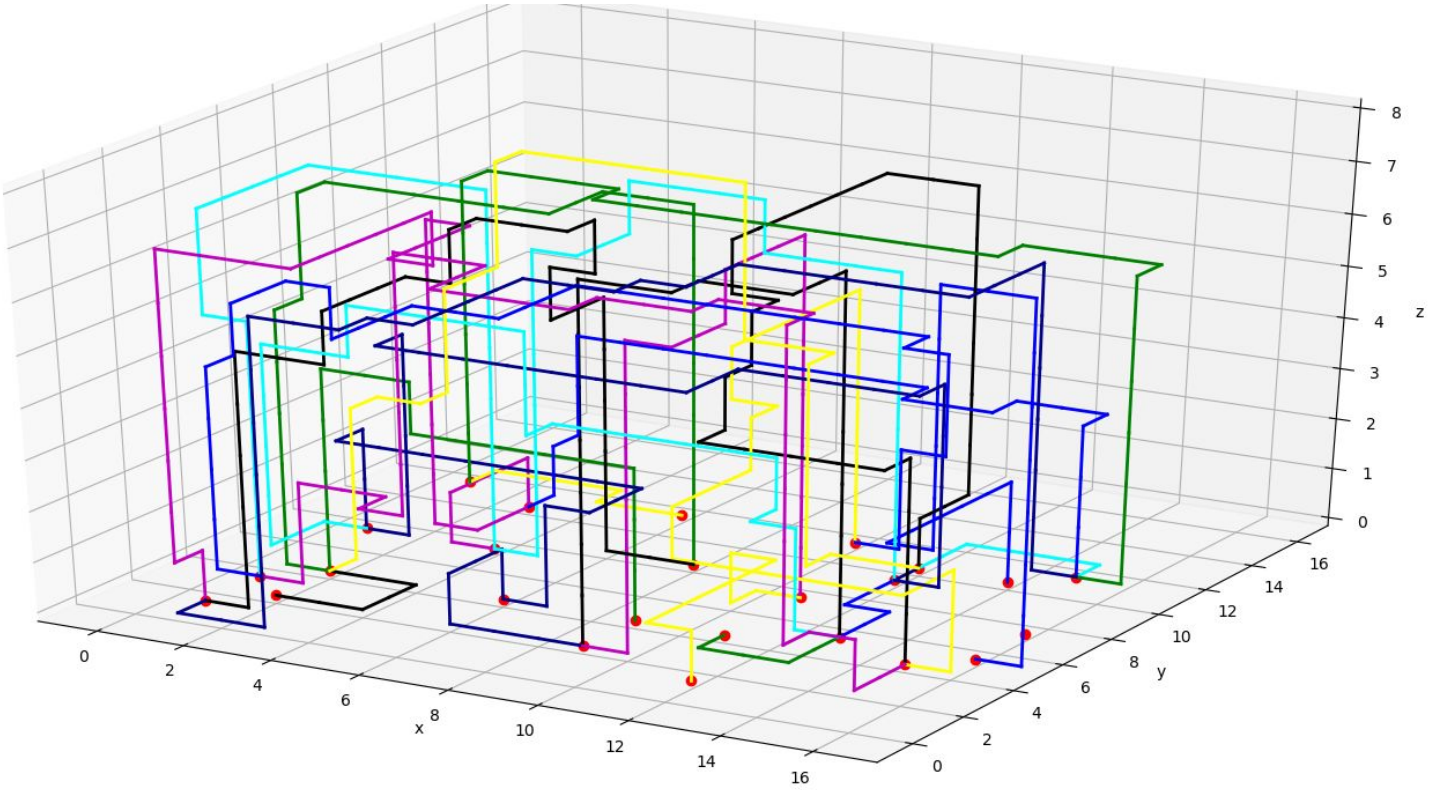
Alleen draadjes met voldoende lengte omhoog



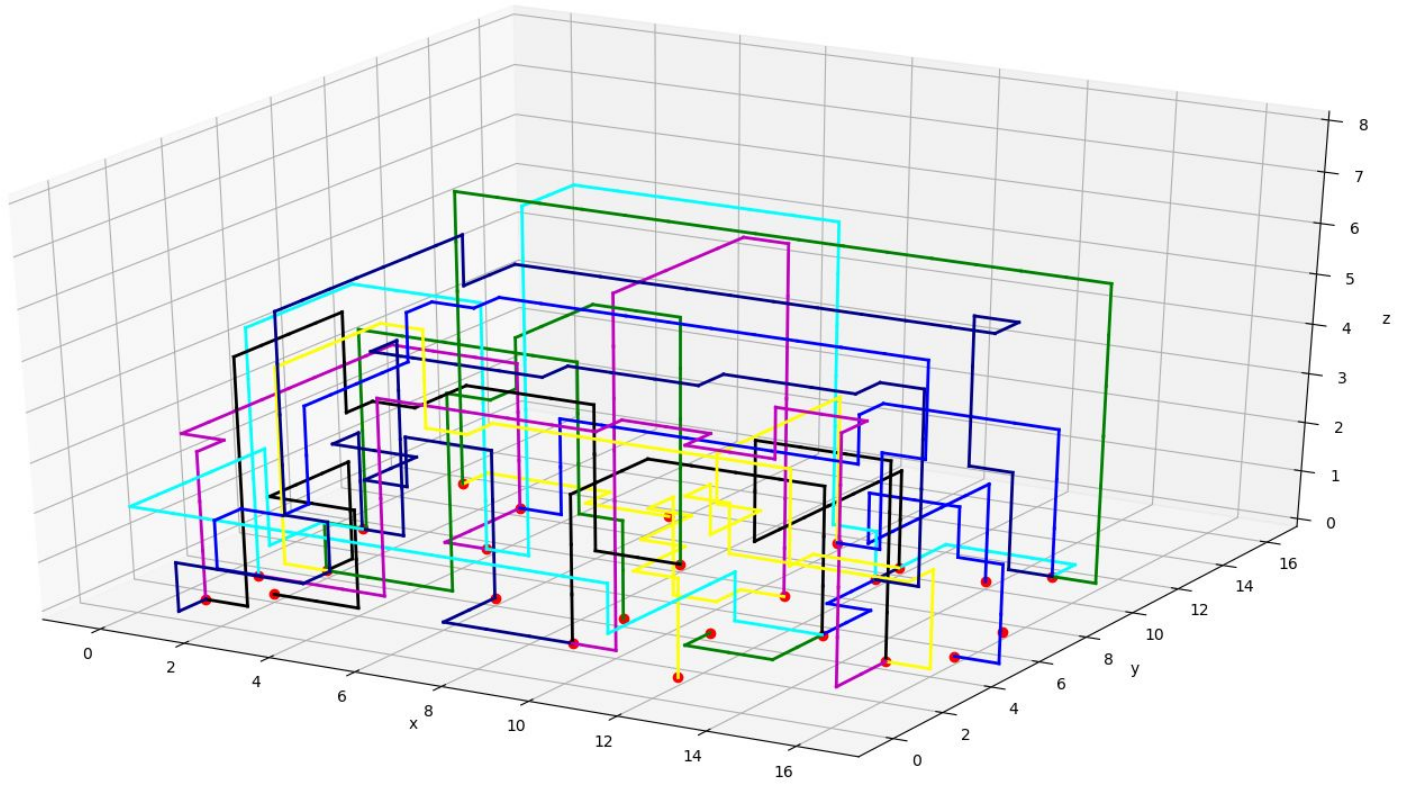
Slide 28 inclusief slide 29



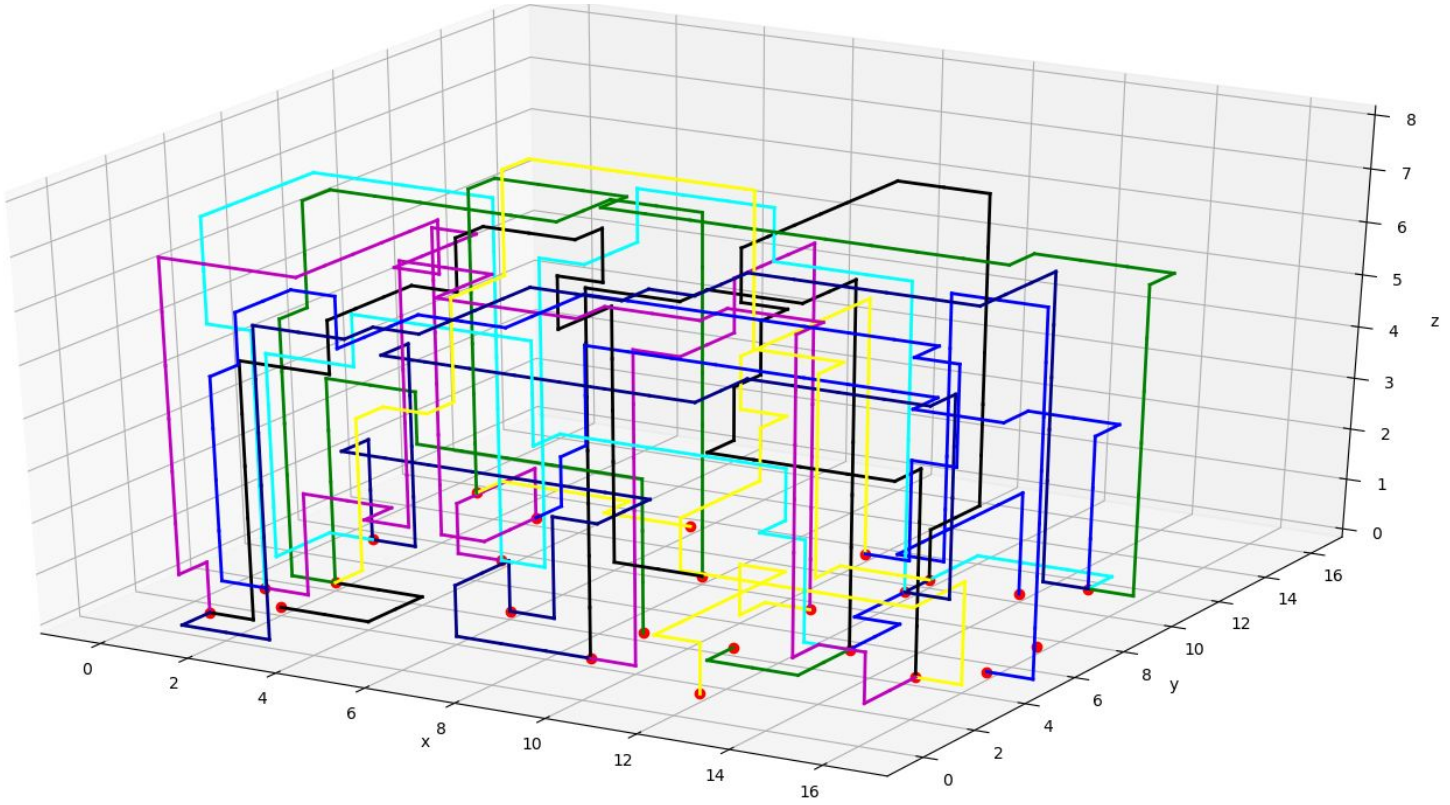
Directe buren van gates vermijden



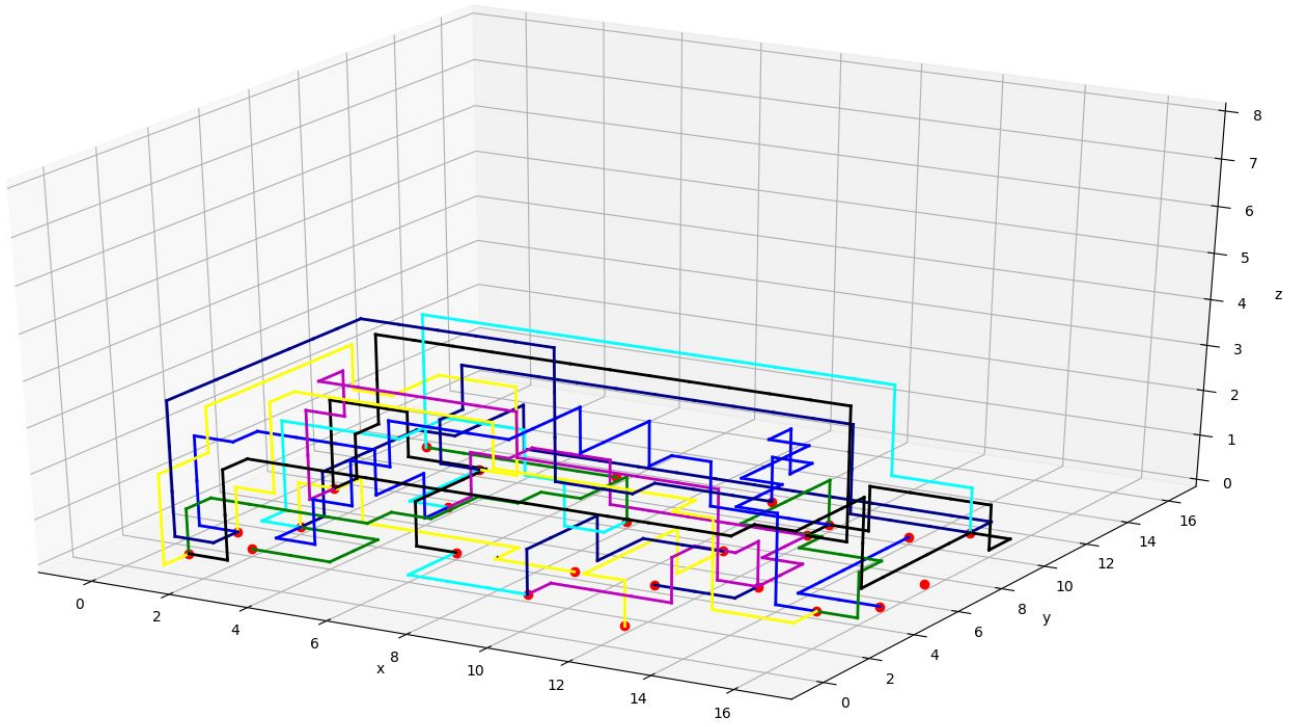
Ook diagonale buren van gates vermijden



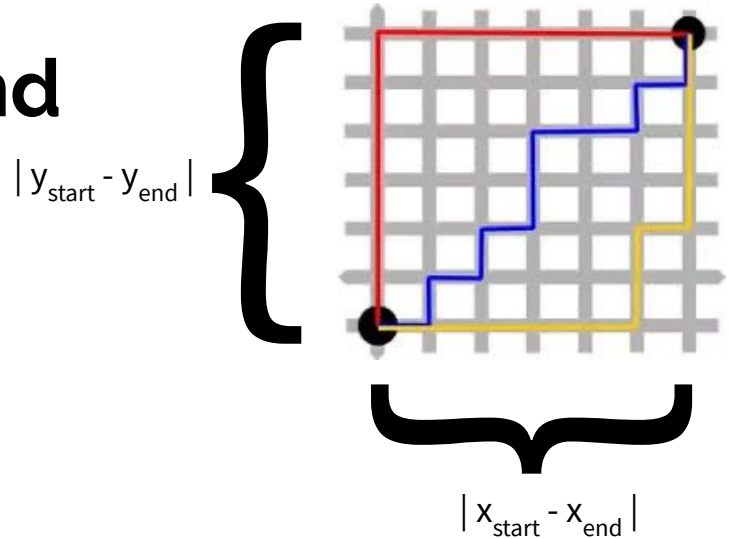
Ook alle z waarden boven gates vermijden



Hillclimber



Extra uitleg bij lowerbound



- Lower bound van de doelfunctie:
  - Voor gate i in gegeven netlist:
    $A_i = |x_{start} - x_{end}| + |y_{start} - y_{end}|$
  - Lower bound = $A_1 + A_2 + \ldots + A_n$
  - Waarbij:
    - i = nummer dat verwijst naar net in netlist (bevat start gate en end gate)
    - n = maximum nummer in netlist (lengte van netlist)
    - x_{start} = x-coördinaat van start gate, x_{end} = x-coördinaat van end gate
    - y_{start} = y-coördinaat van start gate, y_{end} = y-coördinaat van end gate
    - A_n = kortste afstand tussen gates van net n (“Manhattan distance”)
  - Lower bound: wirelength 291 (voor de netlist met 25 gates en 30 connecties)

Toekomstig werk

- Algoritmes werkend op grotere netlist en vergelijken
- Meer onderzoek doen naar andere netlists
- Simulated annealing