**Fruitvliegen**

Algoritmes die sorteren middels inversie

Remco Blom

Hogeschool van Amsterdam

Niels Pannekeet

Hogeschool van Amsterdam

Renske Talsma

Vrije Universiteit

**1. Inleiding -------------------------------------------------------------------------------**

Fruitvliegen zijn kleine, tweevleugelige insecten die zich met name met suikers voeden. Het probleem in deze case heeft betrekking op twee soorten fruitvliegen, namelijk de Drosophila Melanogaster en Drosophila Miranda. Van deze fruitvliegen is veel over het genoom bekend. De twee soorten hebben identieke genen in het genoom, maar deze staan op een andere volgorde. Mutaties in het genoom van de vliegen vinden alleen plaats doordat series genen in hun geheel omkeren: inversie. In de onderstaande figuur is te zien hoe de genomen van respectievelijk Drosophila Melanogaster en Drosophila Miranda zijn opgebouwd. Voor deze case is het doel om uit te zoeken op welke manier de Drosophila Melanogaster in de Drosophila Miranda kan zijn veranderd middels mutaties, en wel in zo min mogelijk inversie stappen en met de minste verplaatste genen.



Figuur 1: De genomen van Drosophila Melanogaster en Drosophila Miranda

Doordat het genoom van de fruitvliegen 25 genen bevat, is het aantal mogelijke manieren waarop deze gesorteerd kunnen staan 25! ofwel 1,5×1025. Het maximale aantal inversies dat nodig is om een elke mogelijke reeks van 25 genen goed te zetten is 24. Het minimale aantal mutaties dat nodig is voor de vlieg is 9. Dit getal is gebaseerd op de elementscore gedeeld door 2. Het aantal elementen zal per inversie idealiter met twee elementen dalen, waardoor de elementscore gedeeld door 2 dus de uiteindelijke minimale hoeveelheid inversies oplevert. De elementscore is een waarde die wordt gebaseerd op het aantal elementen dat zich in een genoom bevindt. Eén element is ofwel een groepje aaneensluitende getallen op volgorde of precies op omgekeerde volgorde, ofwel een los getal. Hoe lager de elementscore, hoe meer getallen er aaneengesloten staan.

Bij het omzetten van het genoom moet echter wel telkens rekening gehouden worden met het feit dat er alleen middels inversie veranderingen plaatsvinden, waardoor de minimale mutatiegrootte 2 genen bedraagt, omdat er bij een inversie altijd minstens 2 genen betrokken zijn. Op elk gegeven moment kan een inversie op 300 verschillende manieren worden uitgevoerd. Dit brengt het aantal mogelijke manieren om een genoom op de goede volgorde te zetten op 30024 ofwel 2,82×1059, wat uiteraard veel te veel mogelijkheden zijn om door te rekenen. Hoewel er in dit kader van genomen en mutaties wordt gesproken, is er uiteindelijk sprake van een sorteringsalgoritme. Mogelijk kan de hieruit gevonden kennis ook worden gebruikt om andere sequenties van data te sorteren met behulp van inversies.

**2. Methodes ------------------------------------------------------------------------------**

Er zijn voor dit onderzoek in totaal 9 verschillende algoritmes gebruikt. De gebruikte algoritmes zijn FindAndSwap, FindAndSwap Reverse, FindAndSwap LoHi, FindAndSwap Iterative, ChunkSwap, EditStar en BeamSearch. Daarnaast is er middels statistisch onderzoek geprobeerd om te achterhalen welke typen genomen zich makkelijk of juist moeilijk op volgorde laten zetten. Met deze kennis zijn de algoritmes vervolgens telkens verbeterd.

**2.1 FindAndSwap en variaties**

Het FindAndSwap-algoritme zoekt in de rij met genen naar het eerste nummer in de reeks, in dit geval 1, en zet dit op de juiste plaats door de genen tot en met dit getal in hun geheel om te draaien. Variaties op dit algoritme, FindAndSwap Reverse en FindAndSwap Iteratief voeren in de basis dezelfde handling uit, alleen begint de Reverse versie met het goedzetten van 25 en werkt vanuit daar naar 1, van rechts naar links. De LoHi variant begint bij 1 en zet dan 25 goed, dan 2 en 24 en werkt zo van binnen naar buiten. De Iteratieve versie doet vrijwel hetzelfde en zet ofwel eerst 1 of 25 goed, afhankelijk van wat de minste verplaatste genen vereist, en gaat van daaruit verder met de swap die vervolgens de minste genen kost om een gen aan één van de uiteinden goed te zetten.

**2.2 ChunkSwap**

ChunkSwap is gebaseerd op hetzelfde principe als FindAndSwap, maar met een toegevoegde functionaliteit die ervoor zorgt dat “chunks” met aaneensluitende getallen die al op volgorde staan niet uit elkaar worden gehaald, maar behouden blijven. Dit algoritme was de eerste stap naar een echte heuristiek die later de basis zou vormen voor de volgende algoritmes.

**2.3 EditStar en varianten**

Het EditStar-algoritme is geïnspireerd op het A\*-principe en maakt gebruik van de score waarbij het aantal inversies en de hoeveelheid elementen bij elkaar worden opgeteld. Een element is ofwel een groepje aaneensluitende getallen op volgorde of precies op omgekeerde volgorde, ofwel een los getal. Hoe lager het aantal elementen, hoe meer getallen er dus aaneengesloten staan. In het geval van D. Melanogaster zijn er 18 elementen. Als alle 300 mogelijke inversies worden gemaakt, belanden alleen degene die krimpen in het aantal elementen in de priority queue en het archief. In het geval dat het aantal elementen met 1 afneemt blijft de score 18 (want 1 inversie + 17 elementen = score 18) en wordt het genoom achter de andere genomen met 18 gezet. In het geval dat de elementen van 2 afneemt, daalt de score naar 17 (want 1 inversie + 16 elementen = score 17), en belandt het genoom vooraan in de priority queue. Hierdoor worden de genomen die met 2 elementen zijn gedaald als het ware beloond en zijn ze snel aan de beurt in plaats van dat ze midden in de snelgroeiende priority queue verdwijnen. Er is ook een variatie op EditStar, welke als doel heeft om de totale hoeveelheid verplaatste genen zo laag mogelijk te houden. Hierbij wordt na alle mogelijke inversies te zijn doorlopen, de gunstigste inversie gekozen: dit is de inversie waarmee de hoeveelheid elementen zoveel mogelijk is gedaald en de inversiegrootte zo klein mogelijk is. Dit gaat door totdat het genoom volledig is gesorteerd.

**2.4 BeamSearch**

BeamSearch is een algoritme dat deels breadth-first te werk gaat. Met BeamSearch worden alle mogelijke inversies uitgevoerd tot drie lagen diep. Er wordt één genoom geinjecteerd waarna alle 300 verschillende inversies worden uitgevoerd en opgeslagen. Op iedere lijst van 25 getallen zijn 300 inversies mogelijk. Van al deze resultaten worden ook alle 300 inversies uitgevoerd, dit zijn vervoglens 90.000 resultaten die de tweede laag vormen. Van de twweede laag worden worden eveneens 300 verschillende inversies uitgevoerd. Dit zijn totaal 27 miljoen resultaten. Deze resultaten vormen de derde en laatste laag en hiervan wordt één resultaat uitgekozen. Met het uitgekozen resultaat wordt dit proces vervolgens herhaald totdat de reeks volledig gesorteerd is. Het kiezen van een resultaat gebeurt op basis van een scorefunctie. Hierbij wordt een punt gegeven voor iedere stap die een getal verwijderd is van de plaats waar het hoort te staan. Dit betekent dat wanneer alles op de juiste plek staat, de score 0 is. Er wordt bij het kiezen gekeken naar het resultaat dat het dichtst bij 0 zit.

Ook van BeamSearch is een extra variatie gebouwd die zich richt op het zo laag mogelijk houden van de totale hoeveelheid verplaatste genen. Hiervoor is dezelfde scorefunctie gebruikt als (een punt voor iedere plaats dat de genen van hun gesorteerde plek af liggen) maar wordt ook nog een extra punt gegeven voor het aantal verplaatste genen in een inversie. Door het zoeken naar de laagste score in de derde laag, wordt er naar de meest voordelige sortering met zo klein mogelijke inversies gezocht. Als resultaat, 15 inversies met een totale inversie grote van 90 genen.

**3. Resultaten -----------------------------------------------------------------------------**

Voor dit onderzoek waren verschillende doelen van belang, namelijk het behalen van een zo laag mogelijk aantal inversies, maar ook het behalen van een zo laag mogelijk aantal verplaatste genen. Tot slot was ook van belang uit te vinden wat een genoom nu moeilijker of makkelijker op te lossen maakt.

|  |  |  |  |
| --- | --- | --- | --- |
|  | Aantal inversies | Gem. grootte inversies | Genen verplaatst |
| Find&Swap | 18 | 8 | 147 |
| Find&SwapRev | 16 | 10 | 161 |
| Find&SwapIteratief | 17 | 11 | 187 |
| Find&SwapLoHi | 18 | 10 | 182 |
| ChunkSwap | 18 | 7 | 132 |
| EditStar | 13 | 9 | 122 |
| EditStarTotalGenes | 15 | 7 | 111 |
| BeamSearch | 13 | 9 | 116 |
| BeamSearchTotalGenes | 15 | 7 | 90 |

Tabel 1: Overzicht van scores van algoritmes op de fruitvlieg

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Gemiddelde aantal inversies | Minimum aantal inversies | Maximum aantal inversies | Gem. grootte  inversies | Gem. aantal  genen |
| Find&Swap | 21,14 | 15 | 24 | 7,60 | 170,59 |
| Find&SwapRev | 21,14 | 14 | 24 | 7,63 | 170,97 |
| Find&SwapIteratief | 21,09 | 15 | 24 | 8,25 | 183,47 |
| Find&SwapLoHi | 21,15 | 15 | 24 | 7,62 | 170,84 |
| ChunkSwap | 20,91 | 13 | 24 | 7,45 | 165,67 |
| EditStar | 17,26 | 13 | 20 | 9,16 | 164,73 |

Tabel 2: Overzicht van scores van algoritmes op set van 2000 pseudorandom genomen

**3.1 Minimum aantal inversies en genen verplaatst**

In de bovenstaande tabellen te zien hoe de algoritmes scoorden op de hoeveelheid mutaties, gemiddelde grootte van de inversies en totale hoeveelheid verplaatste genen. Het allereerste algoritme uit de serie van FindAndSwap-algoritmes leverde voor de vlieg een minimum van 18 inversies op met 147 verplaatste genen. Verbeteringen en aanpassingen aan dit algoritme brachten dat voor de vlieg tot 16 inversies, maar het totale aantal verplaatste genen ging juist omhoog. Over de gehele lijn werd met random genomen geen significant verschil in performance waargenomen tussen de verschillende algoritmes van de FindAndSwap serie. Ook de totale hoeveelheid verplaatste genen en de gemiddelde grootte van de inversies verschilden voor de random gegenereerde genomen nauwelijks. Met ChunkSwap werd voor het eerst een ander systeem gebrobeerd: dit algoritme in tegenstelling tot zijn voorgangers tracht “chunks” van aaneensluitende getallen intact te houden, waardoor de hoop was dat er minder mutaties noodzakelijk zouden zijn om genomen goed te zetten. Hoewel ChunkSwap voor de vlieg qua totale aantal inversies geen verschil maakte, waren er relatief wel veel minder genen verplaatst bij het goed zetten van het genoom, slechts 132. Dit was veelbelovend voor de random gegenereerde genomen, maar hier bleek de prestatie van ChunkSwap toch niet erg ver onder die van de eerdere serie algoritmes te zitten: gemiddeld 165 genen versus gemiddeld 170 genen. Toch leek kijken naar het behouden van getallenchunks een veelbelovende richting, omdat dit bij de vlieg wel tot een sterke verlaging van het aantal verplaatste genen had geleid.

Om de relevantie van getallenchunks te toetsen is er middels de statistiek gezocht naar een verband tussen deze chunks en de performance van algoritmes. Na het meegeven van een score aan genomen op basis van het aantal getallen dat zich in een chunk bevond, de elementscore, bleek dat er een positieve correlatie aanwezig was tussen de elementscore van een een genoom en het aantal mutaties dat nodig zou zijn om dat genoom met bepaalde algoritmes goed te zetten. (*r* = ,453 p < 0,001) Omdat de elementscore lager is voor genomen met meer chunks aaneensluitende getallen, betekent dat dus dat genomen met minder elementen, en dus meer chunks, over het algemeen sneller goed te zetten zouden moeten zijn. Met deze kennis in pacht is vervolgens een variatie op het A\*-algoritme gebouwd. Dit algoritme draagt de naam EditStar en maakt eveneens gebruik van de elementscore. Met het EditStar-algoritme dat op basis van de elementscore de meest optimale inversie uitzoekt, is een resultaat van 13 inversies behaald. Ook qua totaal aantal verplaatste genen en gemiddelde grootte van de mutaties bleek dit algoritme het redelijk goed te doen: een totaal verplaatst aantal van 122 en een gemiddelde grootte van 9 voor de fruitvlieg.

Bij een test van 2000 genomen scoorde het EditStar-algoritme ook aanzienlijk beter dan de voorgangers op het aantal verplaatste genen. Waar eerdere algoritmes gemiddeld zo’n 21 mutaties nodig hadden om een genoom goed te zetten, had het EditStar-algoritme gemiddeld slechts 17 mutaties nodig om de de 2000 random gegenereerde genomen op de juiste volgorde te krijgen. Ook qua aantal verplaatste genen verplaatste de EditStar marginaal gezien iets minder genen. Bij de gemiddelde grootte van de mutatie scoorde hij echter slechter, met 9 gemiddeld versus ongeveer 7,5 gemiddeld. Een variatie op EditStar toegespitst op het zo laag mogelijk houden van de totale hoeveelheid verplaatste genen, EditStarTotalGenes, wist de hoeveelheid 111 te brengen. Helaas leverde dat wel twee extra inversies op waardoor het totaal voor deze variatie op 15 kwam: uiteindelijk niet goed genoeg voor de laagste hoeveelheid inversies, maar wel een lagere score qua totale hoevelheid inversies. Wegens tijdgebrek was het helaas niet mogelijk om voor met deze variant van EditStar ook 2000 genomen te sorteren, maar het is aannemelijk dat de EditStarTotalGenes net als de reguliere EditStar ook uitstekend presteert op de pseudorandom set.

Tot slot is er ook gebruik gemaakt van BeamSearch. BeamSearch leverde eveneens een resultaat van 13 inversies bij het goed zetten van het vliegengenoom. Daarnaast kwam er ook een zeer lage hoeveelheid totaal verplaatste genen uit, namelijk 116. Doordat deze methode zo snel mogelijk alle vergelegen getallen in de buurt van hun juiste plaats zet, hoeven er al snel geen grote inversies uitgevoerd te worden. Hierdoor blijft het aantal verplaatste genen erg laag en kost het weinig inversies om alles op de juiste plek te zetten. Naast een BeamSearch gefocust op de hoeveelheid inversie is er ook nog een versie ontwikkeld die zich focust op de totale hoeveelheid verplaatste genen en deze zo laag mogelijk tracht te houden. Deze versie leverde een resultaat van slechts 90 verplaatste genen totaal, bij 15 inversies. Omdat BeamSearch een breadth-first gericht algoritme is, was het wegens de grote rekentijd helaas niet mogelijk om deze ook zodanig vaak uit te voeren dat een statistische analyse op de resultaten zinvol was geweest. De lange rekentijd van BeamSearch is eveneens de reden dat het in de gevallen waarin men geinteresseerd is in de laagste mutatiegrootte er beter gebruik gemaakt kan worden van EditStar, omdat deze veel sneller rekent (1 seconde per genoom) en eveneens een zeer goed resultaat levert.

Tijdens het vinden van het korste pad is met twee verschillende algoritmes het resultaat van 13 inversies bereikt. Deze 13 inversies zijn behaald met de EditStar en met de BeamSearch. De stappen die beide algoritmes maken verschillen echter wel van elkaar.

**EditStar BeamSearch**



Figuur 2: Vergelijking tussen EditStar en BeamSearch per doorgemaakte inversie

In de bovenstaande figuur zijn de inversiestappen die de twee algoritmes doorlopen te zien. Het verschil in inversies komt voort uit de manier waarop de algoritmes hun prioriteiten stellen. De EditStar stelt zijn prioriteit puur op het vinden van het kortste pad naar het resultaat. BeamSearch probeert zo snel mogelijk de meest ver gelegen getallen in de buurt van en het liefst op de juiste plek te zetten. Dit betekent dat er in het begin grote inversies voorkomen die gevolgd worden door kleinere inversies. Dit is een groot verschil met de EditStar, welke geen rekening houdt met de huidige posities of de inversielengtes, waardoor het mogelijk is dat er lange inversies blijven plaatsvinden.

**3.2. Wat maakt een genoom moeilijk?**

Naast onderzoek naar algoritmes om de genomen met zo min mogelijk inversies en verplaatste genen goed te zetten, is er ook gekeken wat bepaalde genomen moeilijker of makkelijker maakt. Eerder bleek al dat genomen waarin zich minder elementen bevonden ook minder inversies nodig hadden om goed te worden gezet en vice versa. Hoe meer getallen in chunks staan, hoe minder inversies er dus nodig zullen zijn. In een vergelijking van de elementscore met het aantal nodige inversies bleek er voor de EditStar zelfs een correlatie van ,775 (p = <,001) te bestaan. Dit houdt in dat voor het voor het EditStar-algoritme een stuk gemakkelijker is om een genoom goed te zetten naarmate deze minder elementen en dus meer getallenchunks bevat.

Naast de hoeveelheid elementen bleek ook de relatieve plaatsing van getallen ten opzichte van het midden van invloed op de moeilijkheidsgraad van een genoom. Een relatief goed geplaatst getal betekent in dit geval dat de getallen 1 tot en met 12 aan de linkerkant van het genoom staan en dat de getallen 13 tot en met 25 aan de rechterkant van het genoom staan, maar niet noodzakelijkerwijs in de goede volgorde. Door de algoritmes op zowel een groep van 2000 relatief goed geplaatste genomen als een groep van 2000 volledig pseudorandom gegenereerde genomen te draaien en deze te vergelijken, kon er een vergelijking worden gemaakt. Hieruit bleek dat als een genoom genen bevat die relatief al op de goede plek staan, dit er toe leidt dat er minder grote mutaties nodig zullen zijn om het genoom uiteindelijk goed te zetten. Dit is interessante informatie omdat het de prioriteit waarmee bepaalde inversies worden uitgevoerd kan bepalen. Een inversie die 23 van de linkerkant van het genoom naar de rechterkant verplaatst, kan dan bijvoorbeeld de voorkeur genieten over een inversie die minder genen op hun relatief juiste positie brengt.

In de tabel op de volgende pagina is een overzicht te zien van de pseudorandom en relatief correctere genomen. De maximale hoeveelheid genen die nodig lijkt te zijn om een genoom goed te zetten lijkt zo’n 250 te zijn. Bij het herhalen van de algoritmes op genomen waarbij de getallen relatief al goed stonden, werden veel lagere waarden gevonden, zowel het minimum als het maximum waren ongeveer de helft lager.

**Pseudorandom genomen Genomen relatief**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Grootte  inversies | Aantal genen | Min. genen | Max. genen | Grootte  inversies | Aantal genen | Min. genen | Max. genen |
| Find&Swap | 7,60 | 170,59 | 95 | 250 | 4,36 | 90,50 | 51 | 127 |
| Find&SwapRev | 7,63 | 170,97 | 86 | 248 | 4,37 | 90,89 | 50 | 130 |
| Find&SwapIteratief | 8,25 | 183,47 | 88 | 242 | 6,04 | 122,29 | 51 | 214 |
| Find&SwapLoHi | 7,62 | 170,84 | 109 | 252 | 4,37 | 90,77 | 54 | 127 |
| ChunkSwap | 7,45 | 165,67 | 93 | 250 | 4,33 | 88,17 | 51 | 121 |
| EditStar | 9,16 | 164,73 | 118 | 208 | 6,22 | 103,15 | 47 | 222 |

Tabel 3: Overzicht van hoeveelheid verplaatste genen per algoritme

**4. Conclusie ------------------------------------------------------------------------------**

Uit dit onderzoek is gebleken dat Drosophila Melanogaster middels het BeamSearch algoritme in 13 inversies in 116 verplaatste genen in Drosophila Miranda veranderd kan worden. Daarnaast is bekend dat D. Melanogaster met behulp van een aangepaste BeamSearch in 90 verplaatste genen in D. Miranda veranderd kan worden, maar dan met 15 inversies. Tot slot is er ook meer duidelijkheid geschept over welke kenmerken genomen moeilijk maken: namelijk een hoge elementscore ofwel weinig aansluitende getallenchunks en een gebrek aan relatief juiste plaatsing van de getallen. Een hoge elementscore leidt tot meer benodigde mutaties en een lage hoeveelheid relatief juist geplaatste getallen heeft tot gevolg dat er meer genen totaal verplaatst moeten worden om het genoom te sorteren.

Het is jammer dat niet met absolute zekerheid gesteld kan worden dat de laagste hoeveelheid inversies waarmee D. Melanogaster gesorteerd kan worden 13 is. Ook ontbreekt er vooralsnog een concrete boven- en ondergrens van de inversiegrootte voor pseudorandom genomen. Helaas waren onze algoritmes die de D. Melanogaster sorteerden nog niet geschikt voor de random sample. Toekomstige algoritmes zouden zich kunnen focussen op het bepalen van de absolute ondergrens voor elk algoritme om met behulp van deze kennis een algoritme te bouwen dat met zekerheid de laagst mogelijke hoeveelheid inversies en/of verplaatste genen geeft. Ook het volledige breadth-first doorlopen van het genoom zou zekerheid kunnen opleveren, maar dat zal met de huidige processorsnelheden wellicht nog even duren. Voorlopig kunnen we het minimum aantal inversies dat nodig is om van de D. Melanogaster naar de D. Miranda te komen dus op 13 houden.