Fruitvliegen

Algoritmes die sorteren middels inversie

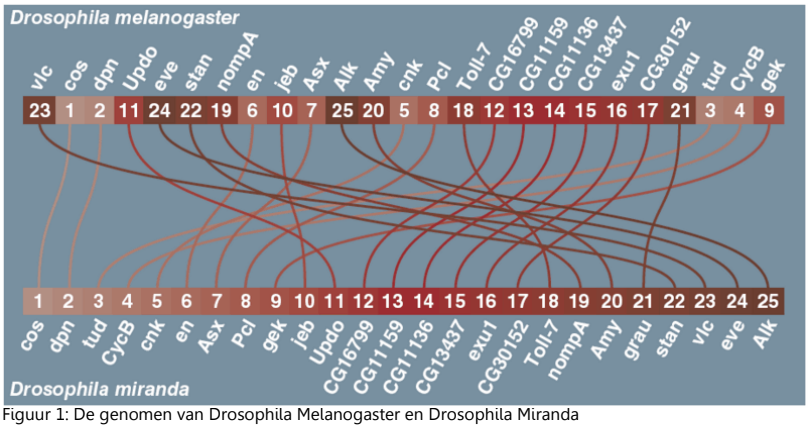
Remco Blom 11079371
Hogeschool van Amsterdam

Niels Pannekeet 11035668
Hogeschool van Amsterdam

Renske Talsma 10896503
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 het 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 inversies, en wel in zo min mogelijk inversie stappen en met de minste verplaatste genen. Daarnaast is het doel om uit te zoeken wat bepaalde genomen moeilijk of juist makkelijk sorteerbaar maakt.



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 geordend kunnen staan 25! ofwel $1{,}5\times10^{25}$. Het maximale aantal inversies dat nodig is om een elke mogelijke reeks van 25 genen goed te zetten is 24 (n-1), omdat met de laatste inversie altijd twee genen goed worden gezet. Het minimale aantal inversies dat nodig is voor de vlieg is 9. Dit getal is gebaseerd op de elementscore gedeeld door 2. 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. Het aantal elementen zal per inversie idealiter met twee elementen dalen, waardoor de elementscore gedeeld door 2 dus de uiteindelijke minimale hoeveelheid inversies oplevert.

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 inversiegrootte 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 300^{24} ofwel $2{,}82\times10^{59}$, wat uiteraard veel te veel mogelijkheden zijn om door te rekenen. Hoewel er in dit kader van genomen en inversies 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 algoritmes gebruikt. De gebruikte algoritmes zijn FindAndSwap, FindAndSwap Reverse, FindAndSwap LoHi, FindAndSwap Iterative, ChunkSwap, EditStar, EditStarTotalGenes, BeamSearch en BeamSearchTotalGenes. 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 handeling 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 buiten naar binnen. 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. 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 het aantal elementen laten krimpen 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 score 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. Van al deze resultaten worden ook alle 300 inversies uitgevoerd, dit zijn vervoglens 90.000 resultaten die de tweede laag vormen. Van de tweede laag worden eveneens 300 verschillende inversies uitgevoerd. Dit zijn totaal 27.000.000 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 (heuristiek). Hierbij wordt een punt gegeven voor iedere

stap dat een getal verwijderd is van de plaats waar het uiteindelijk 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.

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 bovenstaand tabellen te zien hoe de algoritmes scoorden op de hoeveelheid inversies, gemiddelde grootte van de inversies en totale hoeveelheid verplaatste genen. Het eerste 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 naar 161. Over de gehele lijn werd met pseudorandom 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 pseudorandom gegenereerde genomen nauwelijks: de algoritmes van de FindAndSwap-serie scoren allemaal ongeveer gelijk.

Met ChunkSwap werd voor het eerst een ander heuristiek geprobeerd: dit algoritme tracht in tegenstelling tot zijn voorgangers "chunks" van aaneensluitende getallen intact te houden, waardoor de hoop was dat er minder inversies 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 pseudorandom 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 met behulp van 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 inversies dat nodig zou zijn om dat genoom met bepaalde algoritmes goed te zetten. (r = ,453 p<,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. 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 inversies bleek dit algoritme het redelijk goed te doen: een totaal verplaatst aantal van 122 en een gemiddelde inversiegrootte van 9 voor de fruitvlieg.

Bij een test van 2000 genomen scoorde het EditStar-algoritme ook aanzienlijk beter dan de voorgangers. Waar eerdere algoritmes gemiddeld zo'n 21 inversies nodig hadden om een genoom goed te zetten, had het EditStar-algoritme gemiddeld slechts 17 inversies nodig om de de 2000 pseudorandom 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 inversie scoorde hij echter slechter, met 9 gemiddeld versus ongeveer 7,5 gemiddeld. Hoewel de EditStar dus minder mutaties uitvoert, zijn de mutaties gemiddeld wel iets groter.

Een variatie op EditStar toegespitst op het zo laag mogelijk houden van de totale hoeveelheid verplaatste genen, EditStarTotalGenes, wist de hoeveelheid naar 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 groote van 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 lagere hoeveelheid totaal verplaatste genen dan bij de EditStar-algoritmes 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 inversies 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 inversiegrootte er beter gebruik gemaakt kan worden van EditStar, omdat deze veel sneller rekent (1 seconde per genoom) en eveneens een zeer goed resultaat levert.

3.2 Vergelijking kortste paden

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, omdat EditStar en BeamSearch van verschillende heuristieken gebruik maken.

	EditStar	BeamSearch
0	(23, 1, 2, 11, 24, 22, 19, 6, 10, 7, 25, 20, 5, 8, 18, 12, 13, 14, 15, 16, 17, 21, 3, 4, 9)	[23, 1, 2, 11, 24, 22, 19, 6, 10, 7, 25, 20, 5, 8, 18, 12, 13, 14, 15, 16, 17, 21, 3, 4, 9]
1	(23, 1, 2, 11, 24, 22, 19, 20, 25, 7, 10, 6, 5, 8, 18, 12, 13, 14, 15, 16, 17, 21, 3, 4, 9)	[23, 1, 2, 11, 24, 22, 19, 6, 10, 7, 9, 4, 3, 21, 17, 16, 15, 14, 13, 12, 18, 8, 5, 20, 25]
2	(23, 1, 2, 11, 24, 22, 19, 20, 25, 7, 10, 6, 5, 4, 3, 21, 17, 16, 15, 14, 13, 12, 18, 8, 9)	[23, 1, 2, 3, 4, 9, 7, 10, 6, 19, 22, 24, 11, 21, 17, 16, 15, 14, 13, 12, 18, 8, 5, 20, 25]
3	(24, 11, 2, 1, 23, 22, 19, 20, 25, 7, 10, 6, 5, 4, 3, 21, 17, 16, 15, 14, 13, 12, 18, 8, 9)	[23, 1, 2, 3, 4, 9, 7, 10, 6, 5, 8, 18, 12, 13, 14, 15, 16, 17, 21, 11, 24, 22, 19, 20, 25]
4	(24, 25, 20, 19, 22, 23, 1, 2, 11, 7, 10, 6, 5, 4, 3, 21, 17, 16, 15, 14, 13, 12, 18, 8, 9)	[24, 11, 21, 17, 16, 15, 14, 13, 12, 18, 8, 5, 6, 10, 7, 9, 4, 3, 2, 1, 23, 22, 19, 20, 25]
5	(24, 25, 20, 19, 22, 23, 1, 2, 11, 10, 7, 6, 5, 4, 3, 21, 17, 16, 15, 14, 13, 12, 18, 8, 9)	[1, 2, 3, 4, 9, 7, 10, 6, 5, 8, 18, 12, 13, 14, 15, 16, 17, 21, 11, 24, 23, 22, 19, 20, 25]
6	(24, 25, 20, 19, 3, 4, 5, 6, 7, 10, 11, 2, 1, 23, 22, 21, 17, 16, 15, 14, 13, 12, 18, 8, 9)	[1, 2, 3, 4, 9, 7, 10, 6, 5, 8, 18, 12, 13, 14, 15, 16, 17, 21, 11, 20, 19, 22, 23, 24, 25]
7	(1, 2, 11, 10, 7, 6, 5, 4, 3, 19, 20, 25, 24, 23, 22, 21, 17, 16, 15, 14, 13, 12, 18, 8, 9)	[1, 2, 3, 4, 5, 6, 10, 7, 9, 8, 18, 12, 13, 14, 15, 16, 17, 21, 11, 20, 19, 22, 23, 24, 25]
8	(1, 2, 3, 4, 5, 6, 7, 10, 11, 19, 20, 25, 24, 23, 22, 21, 17, 16, 15, 14, 13, 12, 18, 8, 9)	[1, 2, 3, 4, 5, 6, 10, 7, 9, 8, 11, 21, 17, 16, 15, 14, 13, 12, 18, 20, 19, 22, 23, 24, 25]
9	(1, 2, 3, 4, 5, 6, 7, 10, 11, 12, 13, 14, 15, 16, 17, 21, 22, 23, 24, 25, 20, 19, 18, 8, 9)	[1, 2, 3, 4, 5, 6, 10, 7, 9, 8, 11, 12, 13, 14, 15, 16, 17, 21, 18, 20, 19, 22, 23, 24, 25]
10	(1, 2, 3, 4, 5, 6, 7, 8, 18, 19, 20, 25, 24, 23, 22, 21, 17, 16, 15, 14, 13, 12, 11, 10, 9)	[1, 2, 3, 4, 5, 6, 7, 10, 9, 8, 11, 12, 13, 14, 15, 16, 17, 21, 18, 20, 19, 22, 23, 24, 25]
11	(1, 2, 3, 4, 5, 6, 7, 8, 21, 22, 23, 24, 25, 20, 19, 18, 17, 16, 15, 14, 13, 12, 11, 10, 9)	[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 21, 18, 20, 19, 22, 23, 24, 25]
12	(1, 2, 3, 4, 5, 6, 7, 8, 25, 24, 23, 22, 21, 20, 19, 18, 17, 16, 15, 14, 13, 12, 11, 10, 9)	[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 21, 20, 19, 22, 23, 24, 25]
13	(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16 ,17, 18, 19, 20, 21, 22, 23, 24, 25)	[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25]

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, omdat deze geen rekening houdt met de huidige posities of de inversielengtes, waardoor het mogelijk is dat er lange inversies blijven plaatsvinden.

3.3. 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 voor al onze eerste agoritmes dat genomen waarin zich minder elementen bevinden ook minder inversies nodig hebben om goed te worden gezet en vice versa. Deze kennis hebben we ook toegepast op de EditStar. In een vergelijking van de initiële elementscore die meegegeven werd aan de 2000 pseudorandom gegenereerde genomen met het aantal nodige inversies om elk van deze genomen te sorteren, bleek er voor de EditStar een correlatie van ,775 (p =<,001) te bestaan. Dit houdt in dat het voor het EditStar-algoritme een stuk gemakkelijker is om een genoom goed te zetten naarmate deze minder elementen en dus meer getallenchunks bevat. De heuristiek van de EditStar gebaseerd op de elementen werkt dus erg goed. Over het algemeen kan dus aangenomen worden dat een hoge elementscore meer benodigde mutaties tot gevolg heeft.

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 voornamelijk aan de linkerkant van het genoom staan en dat de getallen 13 tot en met 25 voornamelijk 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, 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 inversies en minder verplaatste genen 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 of geen 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, zijn veel lagere waarden gevonden, zowel het minimum als het maximum zijn 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 de gemiddelde waarden van een pseudorandom sample van 2000 en relatief sample van 2000

4. Conclusie

Uit dit onderzoek is gebleken dat D. Melanogaster middels het BeamSearch algoritme in 13 inversies in 116 verplaatste genen in D. 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 inversies 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 pseudorandom sample. Ook van BeamSearch is geen overzicht van de performance op 2000 genomen in verband met de lange rekentijd die BeamSearch nodig heeft om een genoom te sorteren.

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 als dit op grote schaal uitgevoerd moet worden, omdat de rekentijd exponentioneel stijgt met de hoeveelheid of grootte van de reeksen. Voorlopig kunnen we het minimum aantal inversies en de minimale hoeveelheid verplaatste genen die nodig zijn om van de D. Melanogaster naar de D. Miranda te komen dus op 13 en 90 houden.