



СУ "Св. Климент Охридски"
Факултет по Математика и Информатика



ДИСЕРТАЦИЯ

за присъждане на образователна и научна степен "Доктор" по научна
специалност 01.01.12 Информатика

Докторант:

Валерия Николаева Симеонова

**Методи на soft-computing в изчислителната биология:
Асемблиране на данни от геномно секвениране**

Научни ръководители:

Доц. д-р Антонии Попов

Проф д-р Красен Стефанов

Доц. д-р Димитър Василев

2014

СЪДЪРЖАНИЕ

<i>Използвани съкращения</i>	4
<i>Речник на терминологията</i>	5
ПЪРВА ГЛАВА: УВОД	7
1. Увод	7
2. Мотивация за избор на проблем и обект на изследване	10
3. Същност на биоинформатичния проблем от гледна точка на методи и алгоритми (софт-компютинг)	11
4. Използвани софт-компютинг методи в тезиса	11
5. Цели и задачи на изследването	11
5.1. Цели и очаквани резултати.....	11
5.2. Задачи, осигуряващи постигането на поставените цели.....	12
6. Обекти на изследването.....	12
ГЛАВА ВТОРА: МЕТОДИ И АЛГОРИТМИ - ЛИТЕРАТУРЕН ПРЕГЛЕД	13
1. Особеноности на данните от NGS при геномно секвениране.....	13
2. Намаляване на грешките във входните данни	14
3. Съпоставяне на секвенции.....	14
3.1. Видове матрици, използвани при генериране на консенсусни секвенции	19
3.2. Припокриване на секвенции	20
3.3. Динамично оптимиране	21
4. Технология Paired-End Mapping (картиране на свързани краища).....	25
5. Методи за асемблиране	25
5.1. Какво представлява процесът “Асемблиране” ?.....	25
5.2. Предизвикателства пред асемблирането	26
5.3. Алгоритми за асемблиране, използващи графи	29
5.4. Алчни граф-базирани асемблатори	33
5.5. Препокриващи/Подреждащи/Консенсусни асемблатори.....	35
5.6. Приложения на алгоритми с Граф на Де Брюйн	38
6. Методи на размитите множества	49
6.1. Софт Компютинг и Биоинформатика.....	50
6.2. Размити множества (Fuzzy Sets).....	52
6.3. Изкуствени Невронни мрежи (ИНМ)	53
6.4. Генетичен Алгоритъм	59
6.5. Подходи за отчитане на непълнотата на информацията.....	62
6.6. Синергизми	66
ГЛАВА ТРЕТА: ДАННИ, АВТОРСКИ ТЕОРЕТИЧНИ РАЗРАБОТКИ.....	69
1. Постановка на задачата	69
2. ДАННИ	70
2.1. Данни от паралелно геномно секвениране	70
2.2. Биологично представяне на данните	71
2.3. Формати данни от паралелно геномно секвениране	88
3. АВТОРСКИ РАЗРАБОТКИ	90
3.1. Обработка на данните	90
3.1. Непълнота на информацията	90
3.2. Общ модел на алгоритъма за асемблиране, чрез използване на МИРПИ	91
3.3. Статистическо профилиране на данните от паралелно секвениране	92
3.4. Откриване и филтриране на фоновия шум от NGS данни	95
3.5. Изкуствени реферативни “прочити-идентификатори” (ИРПИ)	98
3.6. Методи за генериране на консенсусни секвенции	99
3.7. Конструиране на матрица и алгоритъм при “припокриване на последователности от тип overlap-и ”	100

3.8. Метод за оценка на пътя в графа	103
3.9. Метод за запълване на интервалите в пътищата	105
ГЛАВА ЧЕТВЪРТА: РЕЗУЛТАТИ ВЪЗ ОСНОВА НА РАЗРАБОТЕНИТЕ МЕТОДИ И ПРИЛАГАНЕТО ИМ	107
1. Статистическо профилиране на данните от паралелно секвениране.....	107
2. Построяване на мрежа от изкуствени реферативни “прочити - идентификатори” (МИРПИ) 114	
3. Методи за асемблиране на контигите	116
4. Валидация на обиця модел	117
4.1. Заложена валидация в основния алгоритъм	117
4.2. Допълнителна валидация	118
5. Избрани реализации.....	120
5.1. VBA парсинг код за XML данни, с кодиращи/некодиращи граници.....	120
5.2. Работа със SFF файлове в средата R Project	124
5.3. Генериране на статистическите профили – модул с функции	126
5.4. Генериране на МИРПИ – модул с функции	129
ГЛАВА ПЕТА: ЗАКЛЮЧЕНИЕ	134
1. Дискусия на резултатите	134
2. Приноси.....	136
3. Бъдещи насоки.....	136
3.1. Оптимизация на модела	136
3.2. Потенциални възможности за използване на метода за асемблиране при данни от “de novo” секвениране.....	137
4. Изводи и препоръки.....	138
5. Декларация за оригиналност	139
6. Публикации, участия в научни форуми и проекти, обучения.....	139
6.1. Научни публикации	140
6.2. Участия в научни форуми и проекти	142
6.3. Участия в семинари и учебен процес по време на докторантурата	142
6.1. Обучения по време на докторантурата	143
6.2. Цитирания	143
6.3. Благодарности	143
АПЕНДИКС	144
ИЗПОЛЗВАНА ЛИТЕРАТУРА	144
СПИСЪК НА ФИГУРИТЕ	160
СПИСЪК НА ТАБЛИЦИТЕ	161

Използвани съкращения

№	Английско съкращение	Английски термин	Българско съкращение	Български термин
	RAM	Random Accessed Memory	РАМ	Оперативна памет
	GB	Giga Bytes	ГБ	гига байта
	SC	Soft Computing	СК	Софт-компютинг
	AINN	Artificial Intelligent Neuron Networks	ИИНМ	Изкуствени интелигентни невронни мрежи
	SOM	Self Organizing Maps	СОК	Самоорганизиращи се карти на Кохонен
	SVM	Support Vectors Machines	МПВ	Машини с поддържащи вектори
	ACO	Ant Colony Optimization	АМ	Алгоритъм на мраките
	FS	Fuzzy Sets	РМ, РЛ	Размити множества, Размита логика
	RS	Rough Sets	ГЛ	Груба логика
	GA	Genetic Algorithm	ГА	Генетичен алгоритъм
	ESs	Evolutionary Strategies	ЕС	Еволюционни стратегии
	SA	Simulated Annealing		Пълно симулиране
	PSO	Practicle Swarm Optimization		Оптимизация с интелигентен рояк
	MA	Memetic Algorithms		Имитиращи алгоритми
	TS	Tabu Search		Табу претърсване
	GP	Genetic Programming	ГП	Генетично програмиране
	OLC	Overlap/Layout/Consensus		Препокриващи/Подреждащи/Коинсенусни асемблатори
	DBG	De Bruijn Graph		Граф на Де Брюйн
	QVs	Quality Base Calls Value		Качество на сигналите, определящи нуклеотидната база
	NGS	Next Generation Sequencing	ПГС	Паралелно геномно секвениране
	SNP	Single Nucleotide Polimorphism	ЕВНБ	Единична вариация на нуклеотидна база в ДНК, характеризираща многообразието.

Речник на терминологията

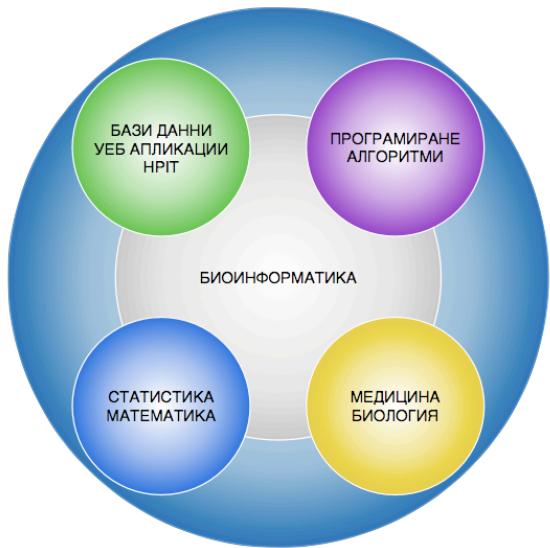
№	Английски термин	Български термин
1	Hash	Хеш
2	K-mer	К-мер
3	Pipeline	Конвейр (IT) – последователна обработка на данните
4	False-Positive	грешен положителен резултат (стат.) – грешка от тип I
5	False-Negative	грешен отрицателен резултат (стат.) – грешка от тип II
6	True-Positive	верен положителни резултат (стат.)
7	True-Negative	верен отрицателен резултат (стат.)
8	Sequencer	Секвенатор (биол.) – машина (платформа), която генерира (чете) ДНК, и произвежда информация под формата на стрингови файлове.
9	Sequencing	Секвениране (биол.) – процес на генериране на файлове, съдържащи ДНК последователности (секвенции).
10	Nucleotide base, Base calling	Нуклеотидна база (биол.) – 4 вида: Adenine (A), Cytosine (C), Guanine (G), Thymine (T). Нуклеотидните бази изграждат ДНК.
11	Sequence	Секвенция (биол.) – поредица от нуклеотидни бази в ДНК, стринг (IT), символен низ (IT) с 4 буквена азбука
12	Short-read	Къс прочит (биол.), секвенция от данни, генериирани с NGS
13	Assembling	Асемблиране (биоинф.). Процес на съединяване на късите прочити в по-големи структури.
14	Contig	Контиг (биоинф.). Структура, получена от асемблирането на къси прочити.
15	Super-contig	Супер-контиг (биоинф.). Структура , получена от асемблирането на контиги.
16	Unitig	Унитиг (биоинф.). Структура, получена от асемблирането на супер-контиги.
17	Scaffold	Конструкция от тип “Скеле” (биоинф.), по която протича асемблирането. Използва се при генерирането на графи.
18	Variation	Вариация в нуклеотидна база (биол.). Може да бъде грешка на секвенирането, мутация или SNP.

Mutation	Мутация (биол.) – еволюционна замяна на една или повече нуклеотидни бази с други.
Homopolymer	Хомополимер (биол.) – поредица от еднакви нуклеотидни бази
Repeat	Повтор (биол.) – секвенция, която се наблюдава на различни места в генома
Chimeric sequence	Химера (биол.) – кодираща ДНК секвенция, която може да бъде открита в различни гени.
Overlap	Презастьпващи (препокриващи) се фрагменти
Reverse sequence	Обратна секвенция, т.е. обръща се реда на четене на фрагмента
Complement Sequence	Комплементарна секвенция (т.е. допълваща до двойна верига) $A \Leftrightarrow T$, $G \Leftrightarrow C$
Reverse Complement Sequence	Обратно комплементарна
3' край в секвенция	Ориентация на секвенирането
5' край в секвенция	Ориентация на секвенирането
Singleton	Единична нуклеотидна база
Purine	Общо наименование за нуклеотидните бази A, G
Pyrimidine	Общо наименование за нуклеотидните бази C, T
De Novo Sequencing	За пръв път секвениране на даден геном
entropия	изразяването на броя на възможните конфигурации или подреждания на градивните частици на системата.
Greedy algorithms	Алчни алгоритми
Strand	Стренд (биол.) – посока на секвениране
Quality Value Base-Calls	Качество на секвенираните нуклеотидни бази (биол.)
Similarity	Подобие (биол.) – сходство, но без доказана обща еволюция
Homology	Сходство (биол.) – дължащо се на обща еволюция
Insertion	Инсерция (биол.) – секвенция, която е вмъкната в друга.
Deletion	Делеция (биол.) – секвенция, която е извадена от друга
Gap	Интервал (биол.) – неустановена секвенция, която принадлежи на друга
Alignment	Сравняване, съпоставяне (биоинф.). Налагане (IT)

ПЪРВА ГЛАВА: УВОД

1. Увод

Изчислителната биология, или още биоинформатика, е мулти-дисциплинарна наука, чиято основна цел е да интерпретира и анализира биологични данни, като се използват изчислителни методи. Тя обединява знания, информация, средства, методи и техники от биологията, компютърните науки, математиката и статистиката, с цел да бъдат анализирани различни видове данни, като най-често това са експериментални и клинични данни, данни от т.н. „омикс“ технологии: геномика, протеомика, метаболомика, транскриптомика и др. Анализът на такива данни е свързан с няколко етапа, включвайки препроцесинг, за минимизиране грешките от генерирането им, както и различни фази на обработка, включващи различни методи, а също така и валидиране. Интерпретацията и на анализите на такъв тип данни е необходима, с цел създаване на нови знания, включвайки отиване на нови биологични единици (гени, протеини, регулаторни елементи), неизвестни до сега биологични принципи, еволюционни тенденции и еволюционна история.



Soft Computing е много подходящ при работа с амбициозни задачи, каквито поставят биологията и протеомиката. Целта на СК е като използва доверителния интервал на несигурността и непълнотата на данните, да апроксимира обосновано данните, така че тази апроксимация да отговаря в

Софт Компютинг е понятие за комплекс от методологии, които работят в повечето случаи съвместно, като по този начин се осигуряват гъвкави решения при обработката и анализа на подадената информация. Този комплекс от

максимална степен на истината. Този резултат трябва да бъде постигнат с минимална цена, както от времева гледна точка, така от от към ресурси. При правилна работа с методите решенията, които се дават са много близки до тези, които човек би взел. Методи на софт-контролинг са: FS, AINN, EAs (GAs, SVM), GP, ES, wavelets, RS, SA, PSO, MA, ACO, TS and теория на хаоса.

"Всеки изчислителен процес, който нарочно включва неточности в изчисляването на едно или повече нива и позволява тази неточност или да се промени (намали) нееднородността на проблема, или да се "омекоти" желаната степен на оптимизация на определен етап, се определя като принадлежащ към в областта на софт компютинга"[1].

Друг начин за дефиниране на СК, е примането за антитеза на това, което бихме могли да наречем хард компютинг. Следователно СК би могъл да се разглежда като поредица от техники и методи, при които реални практически ситуации се анализират и оценяват, както хората се занимават с тях, т.е. въз основа на интелигентност, здрав разум, разглеждане на аналогии, подходи и т.н. В този смисъл, СК е семейство от методи за решаване със специфична резолюция на проблем, начело с апроксимации разсъждения, функционални и оптимизационни методи, включително методи за търсене.

Като биоинформатични задачи могат да се определят: сравняването на секвенции, определяне на структурата на ДНК фрагмент и/или протеин, идентифициране на нов ген (или на неговия промоутър), филогенетичен анализ, анализ на генна експресия, метаболитни процеси, откриване на нови лекарства и др.

За разлика от първите аналитични инструменти в тази област, новите инструменти използват статистиката не като основно средство, а като спомагателно, като ролята ѝ в повечето случаи се свежда до оценка на коректността на резултата - до колкото това може да бъде оценено. Въпреки това, елементът на оценка винаги присъства във всеки един метод и това е едно от съществените различия между различните методи.

Методите на СК са подходящи за работа с биологични данни, защото предполагат обработката на непълни и неточни данни в големи обеми, а точно такива по характер са биологичните данни. Биоинформатиката позволява анализът да бъде върху много обекти едновременно, точно какъвто биологичните данни изискват.

Асемблирането на данни от ПГС е процес на йерархично структуриране на данните, при който те се подреждат, така че да се получи *in silico* реконструкция на изследвания геномен обект. Тази реконструкция е предполагаема, и поради тази причина се прилагат различни методи за оценка достоверността на реконструкцията. При асемблирането група от къси прочити формират т.нар. контиги, а контигите в т.нар. скафолдове (scaffolds), които представляват наредени контиги. Този алгоритъм на асемблиране съдържа различни стъпки, основаващи се предимно на множествено сравняване на секвенции, което е съпроводено с изграждането на т.нар. консенсусни секвенции, които трябва да отразяват общия вид на препокриващите се райони от късите прочити.

Резултатът от асемблирането се оценява чрез големината и точността на контигите и супер контигите. Големината на асемблираните прочити обикновено се определя статистически, като се включват показателите: максимална дължина, средна дължина, комбинирана обща дължина, N50. Показателят N50 всъщност представлява дълчината на най-малкия контиг сред най-дългите контиги, чиято обща дължина образува минимум 50 % от всички асемблирани данни.

Технологиите за ДНК секвениране споделят едно фундаментално ограничение: късите прочитите са много по-малки по размер дори от най-малките геноми. Технологиите за ПГС преодоляват този недостатък чрез използването на различни ИТ решения, главно основани на оптимизационни и скорингови алгоритми, съпроводени с използването на методи на изкуствен интелект и статистическа валидация. Това дава възможност да се използва асемблиращ софтуер, който възстановява структурата на изследвания геном с по-голяма точност.

Пред всеки един софтуер за асемблиране стоят редица предизвикателства, продуктувани от общите характеристики за всички геноми: наличието на хомополимерни участъци, наличието на повтарящи се секвенции, грешки на секвенатора, единични нуклеотидни полиморфизми, които допринасят за разнообразието на гените и тяхното проявление. Високата степен на покритие на генома способства по-лесното определяне и разделяне на повторите, но се влияе негативно от високите нива на грешка от самия процес на секвениране.

Асемблирането на данни от пълното геномно секвениране е съпроводено от комплексността на изчислителния процес на големи обеми от данни.

Обикновено алгоритмите за асемблиране на данни, а така също и техните имплементации се характеризират с висока степен на комплексност и

сложност. Една операция по асемблиране може да изисква високо производителни изчислителни платформи, особено за големите геноми. Успехът на един алгоритъм може да зависи от прагматичен инженеринг и евристики, получени емпирично по правилото на палеца. Евристичният подход може да помогне за преодоляване на сложно повтарящи се модели в реални геноми, случайни и систематични грешки в реални данни, както и при разрешаване на проблема с физическите ограничения на компютри от тип работна станция.

2. Мотивация за избор на проблем и обект на изследване

Биоинформатика е област от науката, която съчетава в себе си нови технологии, нови предизвикателства за обработка на данни със нестандартен формат, като използва методи на софт-компютинг и статистиката, които предлагат различни възможности за оптимизации и решения, основаващи се природни механизми.

В днешно време генетиката произвежда голямо количество данни, характерни със своята комплексност, неточност и обеми. Една и съща задача намира множество реализации, поради факта, че е трудно да се докаже кое решение е най-вярно. Това е област на изследвания, които генерират нови знания и нови теории. Паралелното секвениране на геномни данни все още решава задачата за асемблиране на геном, независимо дали за него съществува или не референтен такъв, който да се възприема за относително верен и точен. Проекти като 1000 генома са заложени в международни програми за секвениране на геноми на различни видове от растителния и животинския свят, както и на морските микроорганизми (проект на Крейг Вентър). Това означава, че в близките поне 10 години ще има нужда от имплементации, методи и алгоритми за асемблирането им, т.к. това е основна задача при паралелно секвенираните данни. Т.к. технологиите от своя страна също се развиват, то вероятността даден организъм да бъде секвениран с различни технологични платформи расте, а с това далеч не се намаляват възможностите за прилагане на асемблиране върху данни, за които може да се приеме, че съществуват референции. Дори напротив – множеството от задачи, поставяни от биоинформатиката расте заедно с технологичния прогрес.

Ето защо за тема на дисертацията бе избрано заглавието “Методи на софт-компютинг в изчислителната биология: Асемблиране на геномни данни”

3. Същност на биоинформатичния проблем от гледна точка на методи и алгоритми (софт-компютинг)

Софт-компютинг е понятие за комплекс от методологии, които работят в повечето случаи съвместно, като по този начин се осигуряват гъвкави решения при обработката и анализа на подадената информация. Този комплекс от методологии е много подходящ при работа с амбициозни задачи, каквито поставят биологията и съвременните й измерения като геномика, протеомика, метаболомика. С други думи СК е група от методи обработващи комплексни задачи, с помощта на различни методи от евристиката, изкуствения интелект, функционални и оптимизационни методи за разпознаване, търсене, валидиране и др.

Използването на СК трябва е икономично на времеви и изчислителни ресурси. По-конкретните задачи на софт-компютинг в изчислителната биология или биоинформатиката, обхващат: сравняването на секвенции, определяне на структурата на ДНК и/или протеин, идентифициране на нови ген, асемблиране на данни от паралелно секвениране, филогенетичен анализ, анализ на генна експресия, метаболитни процеси, компютърен дизайн на нови лекарства и др. Методите на СК са подходящи за работа с биологични данни особено защото предполагат обработката на непълни и неточни данни в големи обеми, като анализът да бъде върху много обекти едновременно и паралелно, а точно такива по характер са молекулярните данни от съвременната изчислителна биология.

4. Използвани софт-компютинг методи в тезиса

В настоящата дисертация е използван комплекс от методи, включващ:

- Статистически анализ
- Невронни мрежи
- Синергизъм между Граф на Де Брюйн и OLC

5. Цели и задачи на изследването

5.1. Цели и очаквани резултати

Основната цел на настоящата работа е разработването на метод и тестване на варианти за алгоритмизации, решаващи подзадачите на метода за асемблиране на данни от паралелно секвениране, за които съществуват референтни секвенции.

5.2. Задачи, осигуряващи постигането на поставените цели

За постигането на така поставените цели се наложи да се решават следните по-специфични задачи:

- Дефиниране на задачата за анализ, с ясно определяне на ограниченията които се налагат от характера на данните
- Определяне на общата схема от проблеми, които следва да се предложат методи и алгоритми, заедно с връзките и преходите между тях. Всеки проблем съставлява отделна съвкупност от алгоритми, която може да се изпълни и самостоятелно върху подходящ тип данни:
 - Проблем 1: генериране на предварителни статистики за данните
 - Проблем 2: откриване и корекция на фоновия шум в NGS данните – целта е да се подобри качеството на използваните данни. Третира се като опция. В основата на метода стои невронна мрежа, която използва както генериирани данни от Проблем 1, така и данни от сравняване.
 - Проблем 3: генериране на т.нар. МИРПИ – Мрежа от изкуствени референтни прочити-идентификатори. Тази мрежа се изгражда въз основа на референтния геном. Присъствието ѝ се обосновава от замисъла за създаване на последователна мрежа от къси прочити, мястото между които следва да се запълни от асемблатора.
 - Проблем 4: да се разработи матрица, която да се използва при сравняването на секвенциите, и която да дава необходимите резултати при “препокриващи се секвенции”
 - Проблем 5: асемблиране на данните от паралелно секвениране, базиран на синергизма OLC - графи на Де Брюйн.
 - Проблем 6: предложение за статистическа валидация на получените резултати от асемблирането – тест за достоверност и сравнителен анализ с референтния геном и други платформи за асемблиране на данни от паралелно секвениране

6. Обекти на изследването

В дисертацията са използвани данни от Sanger и 454 за моделното растение *Arabidopsis Thaliana*. Особеното при NGS данните е, че те са за модифициран организъм чрез *Ti Plasmid* с линията *pBII121* (чийто геном е известен и също участва в процесите по обработка), а секвенирането е от тип EST, т.е. секвенираните данни отразяват само кодиращите участъци от ДНК-то на обекта и съдържат общо 79990 символни низа.

1. Особености на данните от NGS при геномно секвениране

През годините са създадени различни технологии на секвениране, като за първото поколение е възприето да се казва, че това е традиционният начин, при който получаваме секвенции на отделни гени от съответния геном. Този тип секвениране се оказва, че е доказано най-точният и в същото време най-бавният и скъп процес в сравнение със последвалите поколения секвенирния (2-ро, 3-то и 4-то), които за краткост ще наричаме “Следващо поколение секвениране” или „паралелно секвениране”, въпреки че в литературата възприетият термин “Next Generation Sequencing” се отнася предимно за втората генерация машини, които предлагат именно това паралелно секвениране. Характерното за всички “следващи”, е че произвеждат множество от секвенции с определена дължина, която е предмет на конкретната технология.

От гледна точка на представянето на данните от секвениране, независимо от технологията, която е използвана за получаването им, това са файлове с буквени и/или цифрови последователности. Азбуката е 4-буквена като е възможно наличието на 5-та буква, която отразява несигурността в конкретната база, т.е. не може да се каже със сигурност коя от 4-те нуклеотидни бази е представена.

Пример | A T G C A A A A C A T G C
 | A T G N N N N N C A T G C



Съществува множество от файлови формати, разработени конкретно за данни от секвениране на ДНК/РНК. В настоящата работа са използвани файловите формати, показани на фиг.1

Фигура 1: Файлови формати за данни от ДНК секвенции

Различни са мотивите, поради които в определени ситуации се предпочита един вид файлов формат пред друг. В следващата таблица са дадени описанията им, както и какви данни съдържат.

2. Намаляване на грешките във входните данни

3. Съпоставяне на секвенции

Методите за съпоставяне на секвенции никога не са били еднозначни не само поради големият обем налична информация, а основно заради промените в секвенциите на ДНК и белтъците в процеса на еволюция. Милионите години на еволюция на живата природа предизвикват значителни различия между съвременните организми в техните геноми (ДНК). В резултат на мутации и селекция се променят отделни нуклеотидни бази, а от там и аминокиселините, които те кодират; променя се дължината на секвенцията в резултат на инсерции и делеции. По-сериозните промени включват сливането на секвенциите на два различни гена.

Подобни промени в секвенцията и дължината на гена могат да прикрият сходството му с подобни секвенции. Съпоставянето на секвенциите една спрямо друга е начинът за разкриване на подобието им. Методите за съпоставяне на секвенции съставят ядрото на много софтуерни инструменти, конструирани за търсене в бази от данни.

Същността на съпоставянето е в локализирането на еквивалентни области в две или повече секвенции така, че сходството между тях да е максимално. В резултат на мутации обаче дори секвенциите на един и същи фрагмент от генома на два близкородствени вида рядко съвпадат напълно. В идеалният случай при съпоставянето на секвенции, ако те произлизат от общ предшественик, се сравняват нуклеотидните бази/аминокиселините, наследени от този предшественик. Без информация за обратното това се постига най-успешно чрез максимализиране на подобието между съпоставяните области.

За илюстрация на този принцип може да послужи сравняването на две кратки, примерни секвенции. Ако се съпоставят така, че да съвпаднат максимално голям брой бази, се получава:

A	C	C	G	G	T	A	T	A	A	A	T	T	C	G	G	G
A	C	C	G	T	G	A	T	A	A	A	T	T	C	G	G	G

Съвпадащите бази в примера са отбелязани с уебелен шрифт. При такива кратки секвенции лесно се вижда, че съпоставянето ясно показва голямото сходство помежду им.

Когато обаче секвенциите се различават в по-голяма степен, съпоставянето им е по-трудно. За пример може да се даде сравняване, при което, като във втората секвенция поради мутация има инсерция на три допълнителни бази. Директното съпоставяне на тези две секвенции, както в предходния пример,

води до загуба на голяма част от подобието, което видимо съществува между тях. Освен това разликата в дължината им води до отчитане на фалшиви съвпадения между несъответстващи позиции.

A	C	C	G	G	T	A	T	A	A	A	T	T	C	G	G	G	-	-	-
A	C	C	G	T	G	C	G	A	A	T	A	A	A	T	T	C	G	G	G

За преодоляването на този проблем в едната или и в двете секвенции се включват празни места, за да се запази максималното подобие между тях.

A	C	C	G	G	T	-	-	-	A	T	A	A	A	T	T	C	G	G	G
A	C	C	G	T	G	C	G	A	A	T	A	A	A	T	T	C	G	G	G

В такъв случай при съпоставянето на две секвенции е възможен повече от един вариант, като най-добрият не винаги е най-очевиден. Това важи с особена сила, когато секвенциите нямат голямо сходство. Затова в основата на методите за съпоставяне на секвенции и претърсване на бази от данни стоят алгоритми за даване на количествена оценка на всяко от възможните съпоставления, на базата на която се филтрират незадоволителните такива според предварително зададени критерии.

Съпоставянето може да покаже степента на хомология между секвенциите. При всички методи за сравняване на секвенции стои фундаменталния въпрос дали наблюдаваните подобия са случаи и следователно с незначително биологично значение или са резултат от общия им произход от еволюционно предшестваща ги секвенция и следователно са хомологни. Термините "хомология" и "подобие/сходство" често са взаимозаменяеми, но имат различно значение. Подобието е описателен термин, който указва само, че въпросните секвенции съвпадат в определена степен, докато хомологията има съществено еволюционно и биологическо значение. В контекста на молекулярната биология хомологията се отнася специфично само за подобия в секвенцията или структурата, дължащи се на общ произход. Следователно хомологни се наричат гените, произлезли от общ ген-предшественик. По време на еволюцията техните секвенции натрупват различия, дължащи се на акумулирането на различни мутации.

Секвенциите могат и да бъдат изключително близки, без да са хомологни, когато са резултат от конвергенция. Конвергенцията обаче рядко дава секвенции с голямо подобие и голяма дължина.

Анализът на резултатите от претърсването на бази данни чрез съпоставяне на секвенции следва да отчита не само горните примери, но и фактори като типовете мутации, които се появяват в еволюцията, разликите във физикохимичните особености на аминокиселините и ролята им в определянето на структурата и функцията на белтъците. Еволюционните процеси, които оказват влияние на различията в секвенциите, трябва да могат да бъдат

изразени количествено и съответно оценявани. Такива схеми на оценяване могат да бъдат включени в алгоритмите за подбиране на най-добрия възможен резултат от съпоставянето на секвенции. Следователно съпоставянето, което според алгоритъма за оценка дава най-висок резултат, се определя като оптимално, а близките по резултат до него се наричат субоптимални.

Най-простият начин за количествено оценяване на подобието между две секвенции е степента на идентичност, която се измерва в проценти. Идентичността описва доколко две или повече секвенции съвпадат във всяка позиция и се измерва, като се преброят съвпадащите бази/аминокиселини, общият им брой се раздели на дълчината на разглеждания участък и резултатът се умножи по 100. В посочения по-горе пример при съпоставянето на секвенциите ACCGGTATAAATTCTGGG и ACCGTGCGAATAAATTCTGGG степента на идентичност ще бъде 75% (15 съвпадения във фрагмент с дължина 20 позиции, включително празните места).

Поради крайния брой бази (4) или аминокиселини (20) винаги съществува вероятност дори при съпоставянето на напълно различни секвенции да има известен брой позиции на съвпадение.

За това има значение и дълчината на секвенциите - една и съща степен на идентичност, изчислена за много дълги участъци, може да се дължи на случаини съвпадения, но при къси фрагменти е от значение.

Точковата матрица е метод за визуално оценяване на подобието между две секвенции и също се базира на идентичността. При този начин на съпоставяне едната секвенция се записва вертикално, а другата - хоризонтално, като базите на всяка позиция представляват отделен ред, съответно колона. Всеки ред се сравнява с всяка колона, като местата на съвпадение се отбелязват с точка. При най-простата система за оценяване съвпадащите позиции се оценяват с 1, а несъвпадащите - с 0, като точките са на всяка позиция с резултат 1.

Степента на идентичност е най-простият и бърз метод за количествено оценяване на качеството на съпоставяне, но не винаги е достатъчен за цялостно представяне на степента на сходство между две секвенции, особено когато се отнася за белтъци. Оценяването на идентични позиции като 1 и на неидентични като 0 пренебрегва значението на типа на разглежданите нуклеотидни бази.

За да има значение степента на идентичност за оценяване на хомологията между две секвенции, тя трябва да надхвърля определена минимална стойност. Статистическите анализи на резултатите от милиони съпоставяния на белтъчни секвенции сочат, че при 90% от двойките структурно близки белтъци степента на идентичност за цялата верига е по-голяма или равна на 30%. При степен на идентичност 25% се оказва, че само 10% от тези двойки показват хомология. Поради тези резултати степента на идентичност в границите 20-30% се нарича

“зона на здрача”, тъй като не е достатъчно доказателство за наличие или липса на хомология, без да се използват и други методи.

Съществуват много различни начини за оценка на съпоставяне. Функцията на всички тях е да дадат единична, количествена стойност, която да отразява степента на подобие или различие между две секвенции. Повечето методи изчисляват сходството и при тях най-високият резултат е най-добър. По-малко са тези, които използват оценка, базирана на разликите. Такава оценка се нарича дистанция и колкото по-малка е тя, толкова по-близки са сравняваните секвенции. Това се използва предимно от приложения за създаване на филогенетични дървета, а измерването на разликите между две хомологни секвенции от различни видове се нарича генетична или еволюционна дистанция.

При съпоставянето на секвенции оценката на всяка двойка нуклеотидни бази се определя от матрица на заместванията, която дава стойности от съпоставянето на всички възможни двойки. През годините са използвани различни видове матрици на заместванията. Някои от тях са базирани на теоретични фактори (например броя мутации, необходими за превръщане на една аминокиселина в друга) или на сходство във физико-химичните свойства. Най-успешните обаче използват точни еволюционни доказателства, получени от анализите на многобройни хомолози на добре изучени белтъци от различни видове организми.

Една от широко употребяваните матрици на заместванията е представена за пръв път 1978 г. от Margaret Dayhoff и сътр., и се нарича РАМ-матрица (РАМ - Percent accepted mutation; процент утвърдени мутации). Тя се базира на реални данни, които моделират еволюционен процес. Секвенциите, използвани при генерирането на този тип матрица, са били много сходни, което от една страна позволява лесното им съпоставяне, а от друга осигурява голяма вероятност разликите при съпоставянето да са резултат от единично мутационно събитие, настъпило за кратък период от време. С изследваните секвенции е било построено филогенетично дърво, от което са били изведени отделните мутации и изчислено съотношението на броя замествания на дадена нуклеотидна база към общия брой срещания на същите нуклеотидни бази в цялата секвенция.

Крайната заместваща матрица е съставена от логаритмичните стойности на вероятността за мутация. Превръщането на вероятностите в логаритми има за цел да сведе крайната оценка на съпоставяне до изчисляване на сбор от отделните оценки на всяка позиция, вместо на произведение от вероятности.

Съществуват повече от една такава матрица и всяка от тях съответства на определено количество утвърдени мутации. Това количество се измерва в единици РАМ (процент утвърдени точкови мутации на всеки 100

аминокиселини), откъдето идва и името на матрицата. Една от най-често използваните матрици на заместванията е PAM250, което означава че на всеки 100 аминокиселини се падат средно 250 мутации (следователно много от аминокиселините са обект на повече от една мутация). Друга матрица - PAM120, е съответства на по-малък брой мутации.

Друга матрица на заместванията, която също намира голямо приложение, е BLOSUM (BLOcks of Amino Acid SUbstitution Matrix; матрица на заместванията на блокове от аминокиселини). Тя е по-късна от PAM, разработена е през 90те години на XX в и е базирана на локално множествено съпоставяне на голям брой силно консервативни, къси фрагменти от белтъчни секвенции, взети от базата данни SWISS-PROT. Тези секвенции след това са разделени на групи в зависимост от степента на идентичност помежду им и честотата на заместване за всички възможни двойки аминокиселини са изчислени между отделните такива групи, без да се строи филогенетично дърво. Различни BLOSUM матрици се получават при вариране на зададената степен на идентичност при разпределението на групите

Хомологните секвенции често се различават по дължина в резултат на делеции или инсерции, възникнали при дивергиранието на секвенциите спрямо изходната. Това налага включването на празни места срещу една или няколко позиции при съпоставянето им, които имат за цел да постигнат възможно най-голямо съвпадение. Трябва да се има предвид, че прекалено честото включване на празни места не отговаря на реалността и обезсмисля значението на съпоставянето. За ограничаване на това при всяко добавено празно място се намалява крайната оценка на съпоставянето с определена наказателна стойност. Структурните анализи показват, че в секвенции със структурно значение инсерции и делеции се появяват по-рядко, а инсерциите обикновено са с дължина няколко остатъка. Тази информация може да бъде отчетена при алгоритмите на оценяването, като се зададе по-малка наказателна стойност при удължаването на вече съществуващо празно място, отколкото при включване на ново. Така най-качественото съпоставяне ще бъде това, което получава максимална оценка при минимален брой празни места.

В практиката включването на празни места почти винаги се налага. Най-простият начин за установяване на най-качественото съпоставяне е генерирането на всички възможни варианти, заедно с празните места, оценяването на всеки от тях и избор на този с най-висок резултат. Това обаче отнема изключително много време - например за секвенция с дължина 100 остатъка ще са необходими приблизително 10^{75} съпоставяния.

3.1. Видове матрици, използвани при генериране на консенсусни секвенции

При сравняването на секвенции, алгоритмите използват т.нар. матрици на субституция. Те позволяват да се дефинира степента на подобност между секвенциите. Използват се както вероятности, така и скоринг матрици, като е възможен преход между тях по формулата[2]:

$$r_n(i,j) = \frac{M_{ji}^n}{f_j} = \frac{P_{ji,n}}{f_i f_j}$$

← Отразява вероятността двойката (i,j) да е в резултат на еволюция

$$s_n = \log r_n$$

← Отразява вероятността двойката (i,j) да е в резултат на случайно събитие

$$s_{ij} = 2 \log_2 \frac{q_{ij}}{e_{ij}}$$

← Логаритмична трансформация

← Отразява наблюдаваните честоти за двойката (i,j)

← Отразява очакваните честоти за двойката (i,j)

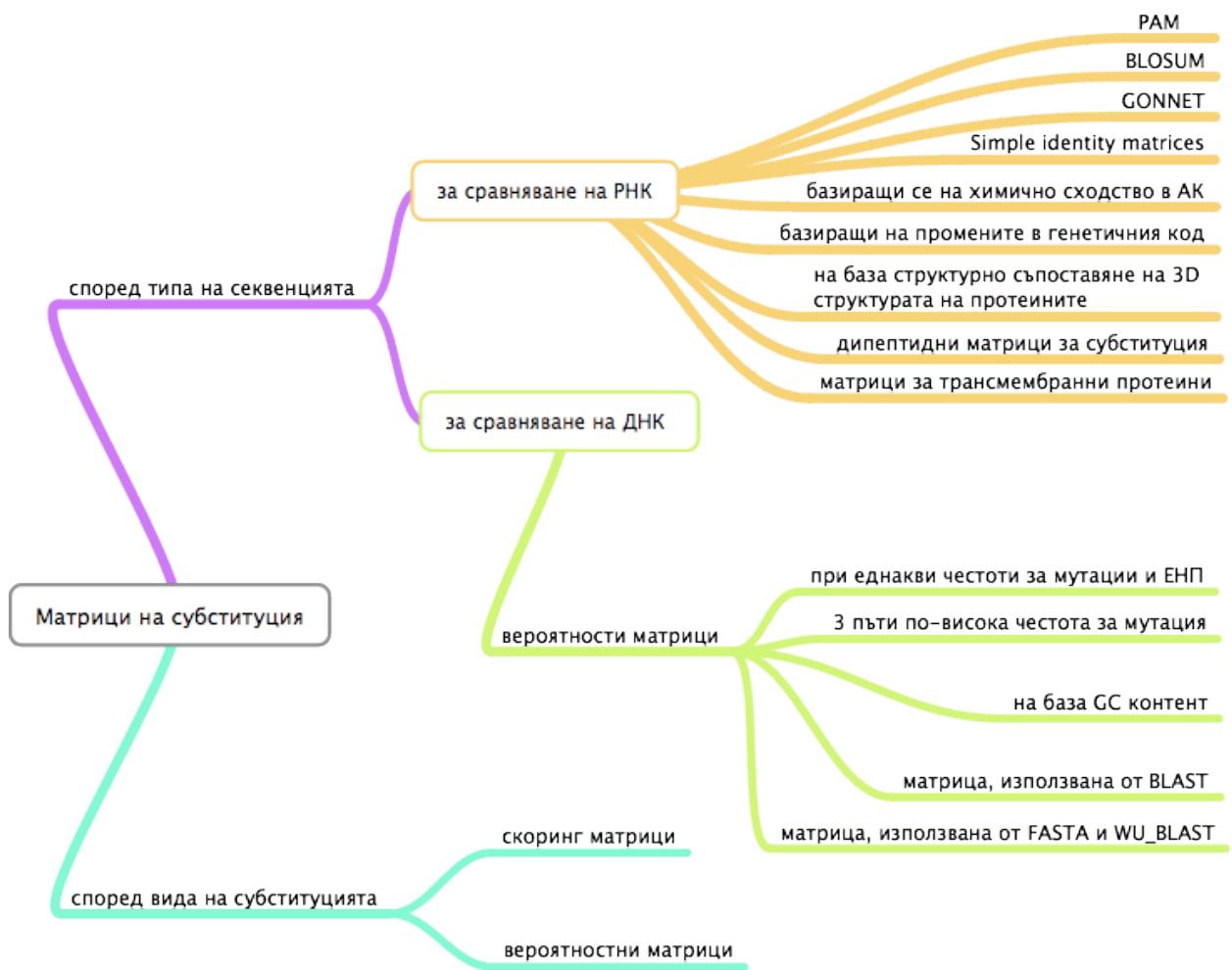
В практиката много по-често се използват логаритмуваните версии на формулите. Положителните страни са, че положителният скоринг обозначава “мутациите”, които приемаме за валидни, а отрицателният – нежеланите.

Скорингът трябва да отразява значимостта на сравняването, възникнало в резултат на еволюционен процес, като взимаме под внимание и това, което можем да очакваме като случайност.

Скорингът трябва да отразява отношението между вероятностите за случайни и неслучайни (еволюционен) модели.

Най-известните матрици са от фамилиите PAM, BLOSUM, FASTA, BLAST. Последните две, условно сме ги нарекли така, т.к. са специфични и се използват само при едноименните алгоритми за локално и глобално сравняване на секвенции. PAM и BLOSUM съдържат индекси в заглавието си. Тяхната съотносимост е обратнопропорционална на индексите им: т.е. ако предполагаме, че секвенциите които сравняваме са близки, то трябва да изберем PAM1 ≡ BLOSUM80, в случай че предполагаме че са далечни, следва да се избере PAM250 ≡ BLOSUM45.[3]. Самите индекси имат следното значение: PAM1 => за всяка позиция, имаме 1 % вероятност да се появи мутация, докато високият индекс при BLOSUM означава и висока степен на консервативност на протеините.

На следващата фигура се виждат различните класификации на матриците.



Фигура 2: Видове матрици на субституция

Алгоритмите за сравняване FASTA и BLAST използват стандартни арбитражни нуклеотидни матрици, които осигуряват съответно 65 и 95% идентичност [4]. Ако дефинираме, че матрицата A(Съвпадение; Несъвпадение), то тези две матрици могат съответно да бъдат описани като $F(5;-4)$ и $B(1;2)$.

3.2. Припокриване на секвенции

Всяка секвенция може да бъде представена като символен низ X , съдържаща n на брой символа и крайна азбука A с дължина r [5]. За дължина на азбуката при геномна секвенция приемаме 4 (A,C,T,G) или 5 (A,C,T,G,N), където (A,C,T,G) са 4-те нуклеотидни бази, кодиращи ДНК веригата, а N отразява ситуацията на невъзможност да се определи със сигурност коя нуклеотидна база е на съответната позиция.

Припокриването на 2 секвенции се дефинира като:

$$O_1 = X_1(A; n) \cap X_2(A; m)$$

където:

A – е крайна азбука

n – е дължината на X_1

m – е дължината на X_2

$O_1(A;l)$, като $l \leq n$ и $l \leq m$

Пример: Ако $X_1=ACTGGGTCA$ и $X_2=TTCAACTGGC$, то $O_2=ACTGG$

Съществуват 3 варианта на припокритие:

- Когато X_1 е припокrita от ляво, респ. X_2 – от дясно
- Когато X_1 е припокrita от дясно, респ. X_2 – от ляво
- Когато X_1 е припокrita в междинен участък, тогава $X_2 \in X_1$

Припокриването на секвенции от гледна точка биоинформатиката се наблюдава като събитие при сравняването на секвенциите. Алгоритмите на съпоставянето се отнасят към динамичното програмиране.

3.3. Динамично оптимиране

Динамичното оптимиране е техника на програмиране, която значително съкращава нужното време за решаване на даден проблем, използвайки вече получени резултати на предходна стъпка от алгоритъма. В повечето случаи тя помага за свеждането на сложността на даден алгоритъм от експоненциална до полиномиална. Т.е. една задача се изчислява не повече от веднъж, ако сме сигурни, че резултатът от изчисленията ще е същият, като резултатът се записва в таблица с вече изчислени стойности, която може да се използва при повторно попадане в същата ситуация.

Приложението на алгоритмите за динамично програмиране в областта на биоинформатиката дава възможност за съсредоточаването на време и ресурси за изследване само на тези съпоставяния, които могат да се считат за оптимални, и избягване на останалите варианти. Динамичното програмиране намира широко приложение в биоинформатиката, при реализирането на задачи като съпоставяне на секвенции, предсказване на протеинови структури, нуклеотидно позициониране, свързване на транскрипционни фактори, класификация на некатегоризирани данни.

3.3.1. Глобално сравняване

Първият алгоритъм, който използва динамично програмиране при съпоставяне на секвенции, е разработен от Saul Needleman и Christian Wunsch, и публикуван

през 1970[6]. Основната идея на този алгоритъм е да направи сравнение между всички възможни двойки остатъци в двете секвенции, което се представя в двумерна матрица, където едната секвенция е записана по вертикалната ос, а другата - по хоризонталната. Всички сравнения между които и да е двойки представляват пътища през така полученото поле, всеки от които може да бъде оценен. Основната идея е с напредване на съпоставянето от единия край на веригата към другия при всяка стъпка да се отхвърлят варианти за съпоставяне, които не дават добър резултат.

Работата на Needleman и Wunsch и до днес стои в основата на много съвременни методи за съпоставяне и търсене на секвенции. Основната разлика е, че при техния метод празните места са обвързани с наказателна стойност, независима от дължината им, докато новите методи използват по-сложни изчисления на наказателните стойности. Запазено е обаче правилото, че празните места никога не се съпоставят помежду си.

Основните принципи, разгледани дотук, могат да бъдат приложени в различни видове съпоставяне в зависимост от естеството на изследването. При две хомологни секвенции, които обикновено са сходни по дължина, съпоставянето покрива целите вериги. Такъв тип съпоставяне се нарича глобално и е най-подходящо за сравняване или търсене на близко родствени секвенции.

Съществуват обаче много други варианти, при които само част от секвенциите са сходни. Такова съпоставяне само на части от секвенциите се нарича локално и намира приложение точно при подобни случаи на търсене на сходни елементи с ограничена дължина в два символни низа. Разчитането само на глобалното съпоставяне не винаги може да разкрие такива малки, но важни участъци на подобие.

3.3.2. Локално сравняване

Локалното съпоставяне е полезен инструмент и при претърсването на бази от данни спрямо непозната секвенция. След идентифицирането на секвенции, които споделят участъци с висока степен на подобие с изследваните, след това между тях вече може да се приложи глобално съпоставяне. Локалното съпоставяне намира приложение и при разпознаване на конкретни функционални участъци.

Широко разпространен алгоритъм за локално съпоставяне е този на Smith-Waterman[7], който е модификация на алгоритъма на Needleman-Wunsch. Вместо цели секвенции, този алгоритъм сравнява сегменти от всички възможни дължини и избира тези, чието сходство е оптимално. Матрицата за оценка в този случай включва както положителни, така и отрицателни стойности като

само съпоставяния, чиято обща оценка е положителна, се разглеждат по-нататък.

Както при глобалното, така и при локалното съпоставяне съществуват методи, разглеждащи само две секвенции (съпоставяне по двойки) или повече от две (множествено съпоставяне). Множественото съпоставяне позволява едновременното сравняване на набор подобни секвенции и има приложение при търсенето на конкретни мотиви, аSEMBЛИрането на NGS данни и др.

3.3.1. Множествено сравняване

Едновременното съпоставяне на голям брой секвенции дава по-точна представа за сходството помежду им от съпоставянето по двойки. То осигурява повече информация за отделните позиции в секвенциите, откъдето може да се съди за цялостно сходство и еволюционни връзки. Това е от голямо значение при използването на методи за съпоставяне на секвенции при генерирането на филогенетични дървета. Множественото съпоставяне илюстрира запазването на консервативните участъци в набора от изследвани секвенции и по този начин помага при идентифицирането на области със структурно и функционално значение.

Съществуват няколко различни метода за множествено съпоставяне. Един от тях е продължение на методите за динамично програмиране, но вместо двумерна матрица, както при съпоставянето на две секвенции, разчита на п-мерна матрица, където p е броят на съпоставяните секвенции. Този подход има доста ограничения, основно поради изискването за голям изчислителен ресурс, поради няма голямо приложение.

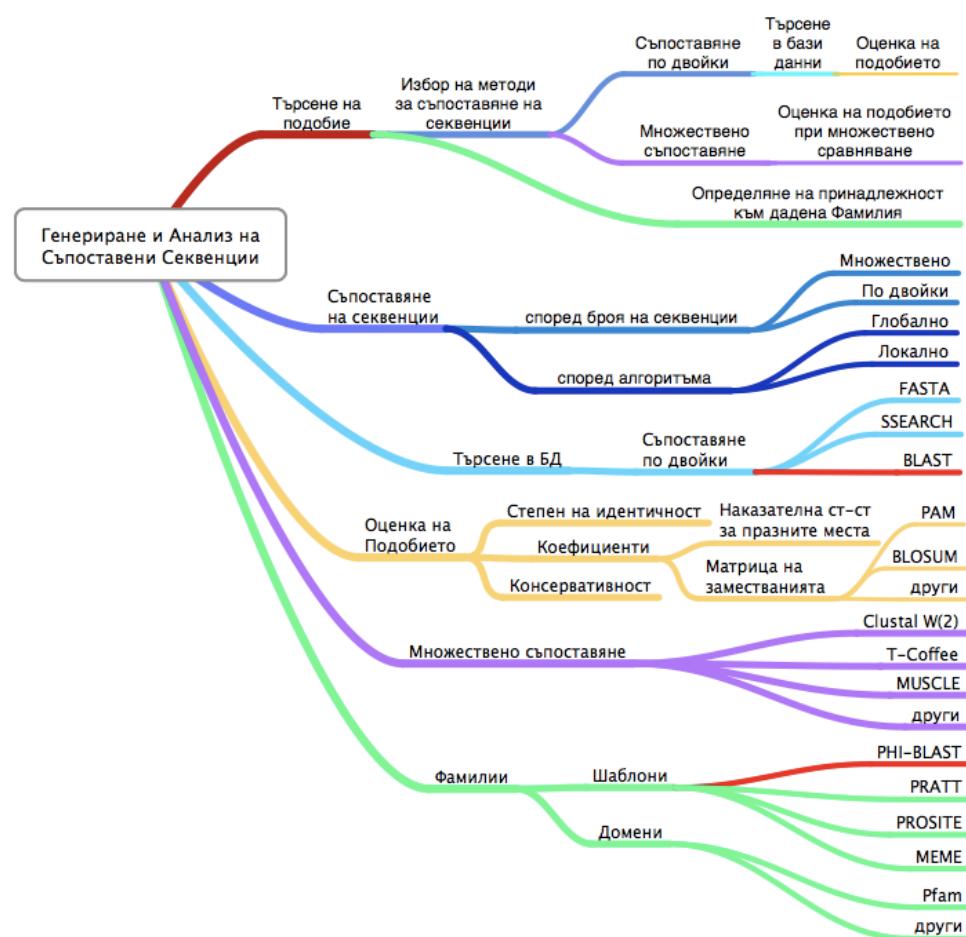
Други методи обикновено също разчитат на динамично програмиране за съпоставяне на двойки секвенции, но използват други техники за комбиниране на резултатите от тях в цялостно множествено съпоставяне. Широко приложение (например в програмата за съпоставяне на секвенции ClustalW, MUSCLE, MAFFT)¹ намира дървото от йерархични методи за множествено съпоставяне. Този метод първо съпоставя по двойки всички секвенции една спрямо друга, след което прави клъстерен анализ на получените данни и генерира йерархия от секвенции според степента на тяхното подобие. Йерархията представлява просто филогенетично дърво и спрямо него се прави множествено съпоставяне на секвенциите, като се започва от най-близкородствените. Макар и относително точен, този метод има и недостатъци, основният от които е, че евентуални грешки при първоначалното съпоставяне

¹ <http://www.ebi.ac.uk/Tools/msa/>

не могат да бъдат коригирани впоследствие, когато се добавя нова информация от следващите секвенции. Този проблем може да се преодолее с включването на стохастични или итеративни методи, например при програмата на Barton и Sternberg[8].

Множествено съпоставяне може да се получи и с помощта на други алгоритми - сегментиран, консенсусен и “разделяй и владей”. Последният от тях се изразява в накъсването на секвенциите на няколко части, за да се намали дължината им, след което се съпоставят получените фрагменти и накрая се сглобяват, за да се получи множествено съпоставяне.

Съпоставянето на две или повече секвенции, независимо глобално или локално, може да бъде подобрено с помощта на допълнителна информация, като например данни за структурните характеристики на една или повече от изследваните секвенции. Често, качеството на резултатите от програмите за автоматизирано съпоставяне зависи от предварително зададени от потребителя параметри, което също може да доведе до получаването на по-достоверна информация.



Фигура 3: Генериране и Анализ на съпоставени секвенции

Целите, с които се прилагат както сравняването на секвенции по двойки, така и множественото сравняване, най-често са две и са отправни точки за последващ анализ и обработка:

- Генериране на сравняване, което описва различните видове несъвпадения, наричани в биологията структурни нуклеотидни вариации. Те включват инсерции, делеции, мутации и едно-нуклеотидни полиморфизми.
- Генериране на консенсусна (т.е. обща) секвенция за всички участници в съпоставянето.

4. Технология Paired-End Mapping (картиране на свързани краища)

Технологията (Korbel and et al. 2007) предполага, че ако имаме дадена секвенция $X_1(A;n)$, то съществуват $L(A;k)$, $R(A;m)$, $M(A;b-a)$, такива че:

1. $k \in [1; a]$, $a < n$
2. $m \in [b; n]$, $b > k$
3. $O_1 = X_1 \cap L = L$
4. $O_2 = X_1 \cap R = R$
5. $M(A; b - a) \in X_1$

Низовете L и R , наричаме “свързани краища”, и съответно са ляв и десен край в секвенцията X_1 , а M е частта от секвенцията X_1 , заключена между L и R .

Тази структура позволява извличането на M , ако познаваме X_1 , L и R .

5. Методи за асемблиране

5.1. Какво представлява процесът “Асемблиране” ?

Асемблирането е процес на юерархично структуриране на данните, при който секвенираните данните се картират, така че да се получи реконструкция на изследвания геномен обект. Тази **реконструкция** е предполагаема, и поради тази причина се прилагат различни методи за оценка достоверността на реконструкцията. При асемблирането група от къси прочити формират т.нар. контиги (по-дълги секвенции, от няколко къси прочита), а контигите в т.нар. скафолдове (scaffolds), които представляват наредени контиги, като на практика процедурата, която се използва за образуването на контигите и скафолдовете в

повечето случаи е една и съща, с тази разлика, че в първия случай обработваните данни са късите прочити, а във втория – скафолдовете. Контигите предполагат провеждането на множествено секвенционно сравняване, което е съпроводено с изграждането на т. нар. консенсусни секвенции, които трябва да отразяват общия вид на препокриващите се райони от късите прочити. Скафолдовете понякога са наричани супер контиги или мета-контиги. Те определят реда на подреждане на контигите, както и тяхната ориентация и интервали между отделните контиги. Топологията на супер контигите може да бъде представена като път или като мрежа. Повечето асемблатори дават като изход освен образуваните супер контиги и неасемблерините или частично асемблираните къси прочити. Най-широко разпространеният файлов формат на асемблирани данни е FASTA, където 4-буквената азбука е разширена и с други символи, отразяващи спецификите от целия процес, като например консенсусните последователности. Тиретата, например, могат да означават бази, които са пропуснати в консенсусната секвенция, но които са били част от късите прочити. Консенсусната верига на Супер контигите може да съдържа знака N в интервалите между отделните контиги, като броят на N може да бъде идентификатор за дълчината на интервала, която се определя на базата на разширени сдвоени краища.

Резултатът от асемблирането се оценява чрез големината и точността на контигите и супер контигите. Големината на асемблираните прочити обикновено се определя статистически, като се включват показателите: максимална дължина, средна дължина, комбинирана обща дължина, N50. Показателят N50 всъщност представлява дълчината на най-малкия контиг сред най-дългите контиги, чиято обща дължина образува минимум 50 % от всички асемблирани данни. Трябва да се отбележи, че статистиките на N50 са сравними само ако пресметнати с една и съща обща дължина. Вариант за сравнение е възможен ако се прибегне до процентно изражение на N50 към общата дължина на най-дългите контиги. Т.к. коректността на асемблирането е трудно измеримо, се прилагат някои техники за измерване на степента на *“mate-constraint satisfaction and violation”*[9].

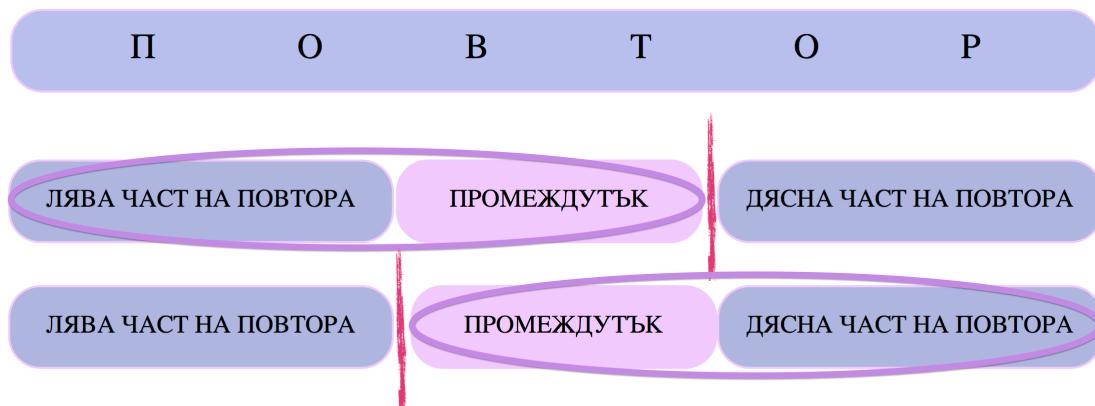
5.2. Предизвикателства пред асемблирането

Технологиите за NGS секвениране споделят едно фундаментално ограничение: късите прочити са много по-малки по размер дори от най-малките геноми. Подходът за Пълно Геномно Секвениране[10] преодолява този недостатък чрез многократно секвениране на целевия геном, но от произволни точки. По този начин се постига по-голяма дълбочина на секвенирането, и се осигуряват множество застъпващи се къси прочити за един регион от генома. Това дава

възможност да се използва асемблиращ софтуер, който възстанови структурата на изследвания геном с по-голяма точност.

Въпреки това, пред всеки един един софтуер за асемблиране стоят редица предизвикателства, продуктувани от общите характеристики за всички геноми: наличието на хомополимерни участъци, наличието на повтарящи се секвенции, грешки на секвенатора, единични нуклеотидни полиморфизми, които допринасят за разнообразието на гените и тяхното проявление. Геномните последователности, които споделят идеални повтори могат да се окажат неразличими, особено ако повторите са по-големи от късите прочити. От друга страна повторите, които не са точни копия, могат да бъдат диференцирани чрез прилагането на сравняване с висока степен на стриктност. Внимателното разделяне на повторите включва корелирането им към шаблони, които отразяват техните специфики[11]. Високата степен на покритие на генома способства по-лесното определяне и разделяне на повторите, но се влияе негативно от високите нива на грешка от самия процес на секвениране. При повтори с висока степен на точност, коректността на отчитането им зависи от т.нар. “промеждутьци”/”интервали”, които представляват уникални единични къси прочити, които разделят повтора, а “промеждутька” остава “залепен” в единния край на една от двете части от повтора, като лявата и дясната част от повтора са идентични.

Повторите, които са по-дълги от колкото късите прочити, могат да бъдат обхванати от анализа чрез т.нар. “промеждутьчни сдвоени краища”, но тогава степента на сложност на задачата рязко се увеличава. За да бъде намерено цялостно решение са необходими два източника: двойки секвенции, които строго да могат да определят границите между ЛЧП/П/ДПЧ и двойка



секвенции, които определят двета края на повтора.

Тези изисквания биха могли да бъдат постигнати за вече секвенирани геноми при някои допълнителни условия. Например, беше показано, че теоретично най-доброто асемблиране на генома на E.Coli от 20 bp несвързани прочита довежда до: 10 % от базите са в контиги с дължина 10 kb или по-голяма, което означава отлично покритие и незначителни грешки [12]. От гледна точка на дълбочината на секвениране и получаване на NGS данни, по-късите прочити не са в състояние качествено да решат проблема с повторите в генома, но пък по-голямата дълбочина на секвенирането увеличава възможността от появата на промеждущни къси повтори.

Т.нар. “резолюция на повтора” е по-трудно определима поради наличието на грешки от секвенирането. От гледна точка на софтуер трябва да се има предвид, че когато резултатът от сравняването на секвенциите не е идеален, това е един вид гаранция, че няма да бъдат пропуснати истинските “напасвания”. Тolerансът в нивата на грешка водят до “false positive joins”. Този проблем съществува особено при прочити от неточни повтори (т.нар. “полиморфни” повтори). False-positive joins могат да предизвикат химерно асемблиране. На практика, толерансът при грешката от секвениране е много важен, за да определянето на точната граница между феномените “грешка” и “полиморфизъм”, това засяга широк спектър от геномни феномени като: полиморфни повтори, полиморфни разлики между неклонални безполови индивиди, полиморфни разлики между нероднински полово диференцирани индивиди, и полиморфни хаплотипове от един неродствен индивид. Ако платформите за секвениране биха могли да генерират прочити при условията на високо покритие и нулема грешка, то тогава софтуерите за асемблиране биха могли да предложат 100 % успешност на процеса, като тогава щеше да остане като по-важен въпрос, бързината на самия процес.

Асемблирането на данни от Пълното геномно секвениране е съпроводено с различно покритие, т.е. с различна дълбочина на секвенирането. Вариацията на покритието се определя от случайността, от вариацията в броя на клетките между източника на ДНК молекули, и зависи от отклоненията, които са характерни за всяка една технология. Много ниската степен на покритие е свързана с проявление на интервали при асемблирането. Вариабилността на покритието проваля валидацията чрез статистическите тестове, базирани върху дълбочината на покритието, а така също и детерминирането на over-collapsed прочитите, чрез диагностика, основаващи се на дълбочината на покритие.

Асемблирането на данни от Пълното геномно секвениране е съпроводено от комплексността на изчислителния процес на големи обеми от данни. За да бъде

постигната по-голяма ефективност, повечето софтуери за асемблиране използват метода на K-mer-ите. Правят се следните допускания: всяка секвенция се състои от K на брой нуклеотидни бази, където K е цяло положително число. В повечето имплементации се използват само последователности от нуклеотидни бази, т.е. така както са наредени и получени при процеса секвениране. Естествено е, прочитите с висока степен на подобност да споделят K-мер-и в районите на припокриване на прочитите. Основната концепция тук е, че е по-лесно да се открият споделените K-мер-и отколкото конкретните райони на припокриване. Бързото дефиниране и откриване на споделените K-мер-и съществено съкращава изчислителното време в сравнение с метода за асемблиране чрез множествено сравняване от типа “всеки срещу всеки”. Компромисът, който се прави при този подход е ниската чувствителност, и поради тази причина някои истински препокривания биват пропуснати. Вероятността едно истинско припокриване да попадне в обсега на споделен K-mer зависи от стойността на K, дължината на припокрития район и нивото на грешка от секвениране. За подходяща стойност на K се приема такава, която е достатъчно голяма, и такава че фалшивите препокривания да не споделят K-мер-и по случайност, и достатъчно малки, за да може да отчете истинските препокривания. Изборът трябва да бъде релативен спрямо вариацията в покритието на прочитите и тяхното ниво на грешка от секвениране.

Обикновено алгоритмите за асемблиране на данни от WGS, а така също и техните имплементации се характеризират с висока степен на комплексност и сложност. Една операция по асемблиране може да изисква високо производителни изчислителни платформи, особено за големите геноми. Успехът на един алгоритъм може да зависи от прагматичен дизайн, евристики, статистически методи, приложения на ИИ. Тези подходи могат да помогнат за преодоляване на сложно повтарящи се модели в реални геноми, случајни и систематични грешки в реални данни, както и при разрешаване на проблема с физическите ограничения на компютърните ресурси. Всичко това създава предпоставки при асемблирането да се използват методи и алгоритми, характерни за СК.

5.3. Алгоритми за асемблиране, използващи графи

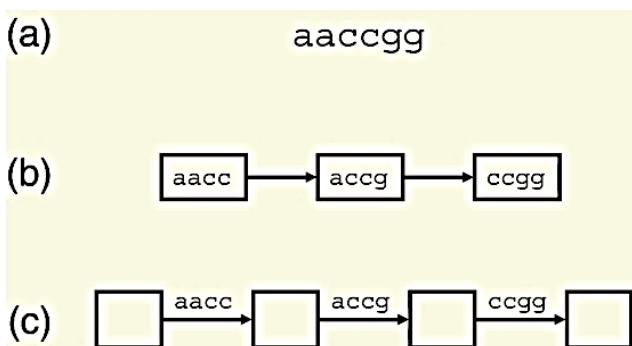
Алгоритмите за асемблиране, използващи графи, биха могли да бъдат групирани в три категории. Групата от методи Overlap/Layout/Consensus (OLC) предполагат т.нар. припокриване на графи. Групата методи, базирани на de Bruijn Graph (DBG) използват някои форми на K-mer графи. Симбиозната група съчетава методи от предходните две категории.

Графът представлява абстрактно представяне на връзките и процесите между различните обекти и е широко използван в компютърните науки. Върховете и дъгите се наричат още възли и рамена, които семантично представляват събития и преходи към тях. В литературата са известни т.нар. насочени графи. Това е случаят, в който рамената имат една единствена посока. Графично един граф може да се представи под формата на кръгове, свързани по между си със стрелки. Значението на стрелките тук е, че показват първоначалното състояние, от което се тръгва и състоянието, до което води. Съвкупността от всички насочени дъги съставя пътя, по който се преминава от състояние в състояние, от връх към връх, като крайното състояние за една дъга винаги се явява начално за следващата. Съществува т.нар. “опростен път”, при който всеки връх участва в общия път само два пъти: един път като начално състояние и един път като крайно състояние, т.е. през него се минава само веднъж в процеса на обход на върховете. Характерно за този тип пътища е, че те не могат сами да пресичат себе си. На върхове и дъгите могат да бъдат присвоени различни атрибути и семантика.

Препокриващите се графи представлят секционните прочити и техните региони на припокриване [13]. Припокритията трябва предварително да бъдат подложени на серия от предварителни изчислителни процеси и сравнявания по двойки. От концептуална гледна точка, върховете на графа са късите прочити, а дъгите - регионите на припокритие. В практиката, графът може да има уникални елементи или атрибути, за да бъдат различими 5' и 3' краишата на всеки къс прочит, права и обратна комплементарност на прочитите, дължина на прочитите, дължина на припокритията, тип на припокритието (suffix-to-prefix или containment). Пътищата на графа показват кои са потенциалните контиги, като всеки път може да бъде превърнат в конкретна последователност от късите прочити. Има два начина, по които могат да бъдат обработени семантиките на двойната спирала на ДНК. Ако графът има отделни върхове за краишата на късите прочити, тогава изходът на пътя в графа трябва да завърши в късия прочит, от който е тръгнал. Ако графът е построен, така че да има отделни рамена за определяне правата и обратна секвенция, тогава пътят за конкретния “стренд” трябва да приключи във връх със същия “стренд”, от който е тръгнато.

Графът на Де Брюйн, който може да се разглежда като подклас на Ойлеров граф, е разработван извън контекста на концепцията за ДНК, с цел да бъдат представени стрингове от букви, принадлежащи на крайни азбуки. Върховете представляват всички възможни стрингове с фиксирана дължина. Дъгите от своя страна са от типа “suffix- to-prefix” и представляват точни припокрития.

Графът от тип K-мер може да се разглежда като частен случай на графа на Де Брюйн. Неговите върхове представляват всички подсеквенции с фиксирана дължина, извлечени от прочита. Неговите дъги, от своя страна са всички региони на припокритие с фиксирана дължина, между подсеквенциите, които са последователни в прочита. Според една формулировка[14], съществува едно рамо за K-мер-а, който излиза от всяка нуклеотидна база, като това не е валидно единствено за K-1 базите. Върховете в този тип графи представляват регионите на припокритие от K-1 нуклеотидни бази. Алтернативно[15], се посочва, че е възможно да има един връх, който да е изражение на K-мер-а, който започва от всяка нуклеотидна база. Докато рамената представляват регионите на припокритие от K-1 нуклеотидни бази. По построение, графът трябва да съдържа пътя, съответстващ на оригиналната секвенция (Фигура 4).



Фигура 4: J.R. Miller et al. / Genomics 95 (2010)
315–327 317

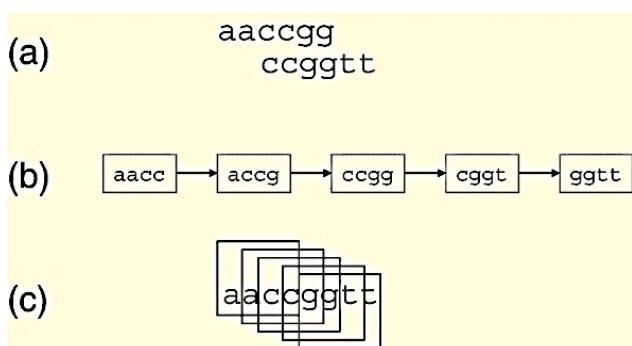
Фигура 4: Прочит, представен в K-мер граф. а) Прочитът е представен чрез два типа K-мер графа с $K=4$. По-големите стойности на K се използват за реални данни. б) Графът има връх за всеки K-мер в прочита плюс по едно насочено рамо за всяка двойка K-мери, които се препокриват в рамките на прочита с $k-1$ нуклеотидни бази. с) Еквивалентен граф има рамо за всеки K-мер от прочита, като върховете имплицитно представлят препокриващите се региони от $k-1$ нуклеотидни бази. В тези примери, пътищата са опростени, защото $K=4$ е по-голямо от 2 bp., колкото е големината на повторите в прочита. Възстановяването на прочита се постига лесно, независимо кой от двата графа се използва.

Пътят пресича сам себе си в елементите от графа представляващи K-мер-ите в секвенциите, с мултиплективен индекс по-голям от единица.

Повтарящият се граф е приложение на K-мер-ния граф [16]. Той предполага по-малка дължина на пътя, т.к. повторите сред прочитите се отразяват веднъж. Върховете и рамената се извличат от асемблираната референтна секвенция. От друга страна уникалните геномни прочити генерират само по един път в графа, докато повторите налагат както конвергенция, така и дивергенция на пътищата, което води до определена цикличност. Повтарящите се графи могат да бъдат използвани за идентификация и каталогизиране на повторите [17].

Графът от тип K-мер може да представя много секвенции вместо една единствена. При прилагането върху данни от WGS асемблиране, K-мер-ния граф е изграден от входни прочити. Всеки прочит генерира собствен път. Прочитите с идеални препокриващи се региони генерираат един и същ път. Ето

зашо, идеалните припокрития лесно се идентифицират без да се налага сравняването на секвенциите (Фигура 5). В сравнение с препокриващите се графи, K-мер графите са по-чувствителни спрямо повторите и грешките от секвенирането.



образуването на контиг, чиято консенсусна използвайки пътя.

Фигура 5: 318 J.R. Miller et al. / Genomics 95 (2010) 315–327

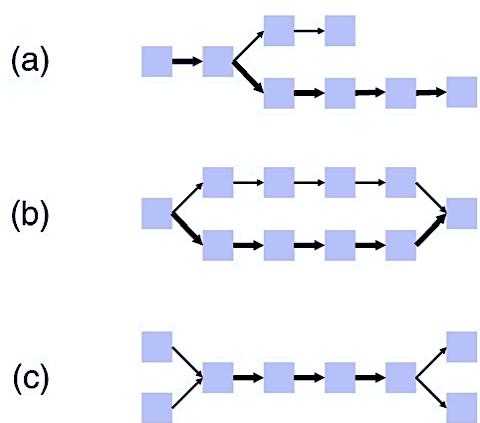
Фигура 5: K-мер граф описва припокриване по двойки. а) Два прочита, несъдържащи грешки в региона на припокриване от 4 нуклеотидни бази. б) K-мер-ен граф, K=4, представя и двата прочита. Сравняването се явява като резултат от конструирането на самия граф. с) Опростения път в графа трябва да доведе до верига лесно може да бъде реконструирана, използвайки пътя.

Всяка единична нуклеотидна грешка в секвенцията води до повишаване на т.нар. “Грешни” върхове в K-мер-ния граф. Всеки такъв връх има шанс да участва в изграждането на пътя и по този начин да доведе до грешна конвергенция на пътищата.

Реалните данни от WGS секвенирането са носители на проблеми при препокриващите се графи и K-мер графите, които могат да се класифицират по следния начин:

1. Spurs - това са кратки “глухи” отклонения от основния път (Фигура 6a). Резултат са от предимно на грешки от секвенирането в единия край на прочита. Могат да бъдат индуцирани също така и когато нивото на покритие падне до нула.
2. Bubbles - това са пътища, които първо се отклоняват, а след това присъединяват. (Фигура 6b). Образуват се когато грешката от секвениране е в средата на прочита, а също така и при единичен полиморфизъм. Качествената детекция на такива пътища не е тривиална задача [18].
3. Усукано въже - това са пътища, които първо са конвергирали, а след това са дивергирали, и образуват визуално усукване (Фигура 6c). Те са резултат от наличието на повтори в таргетния геном.
4. Cycles - това са пътища, които конвергират със самите себе си. Резултат са от наличието на прочити-повтори. Например няколко повтора индуцират появата на малки цикли.

Разклоняването и конвергенцията, разбира се водят до увеличаване комплексността на графа, при което могат да се получат трудни за решение проблеми. В по-голямата си част усложненията са предизвикани от наличието на прочити-повтори и наличието на грешки от секвенирането.



Фигура 6: Сложност на K-мер-ните графи

Фигура 6: Сложността на K -мер-ните графи може да бъде диагностицирана с помощта на информацията за мултипликативността на прочитите. В тези графи, рамената с уделбелени стрелки представляват повторите. а) Наличието една допълнителна база в края на секвенцията предизвиква разклоняване на графа. Същият граф може да се образува и при наличието на нулево покритие след SNP. б) Наличието на нуклеотидна база в средата на прочита, която не се препокрива предизвиква "балон" или алтернативен път. в) Прочитите -повтори водят до т.нар. "усукано въже" чрез системата от конвергиране-дивергирание

Полиморфизмите между хромозоми-донори се очаква да предизвика "балон". В единаква степен дивергенция в пътищата се постига и от прочитите-повтори. в) Прочитите -повтори водят до т.нар. "усукано въже" чрез системата от конвергиране-дивергирание

В контекста на теорията за графите, асемблирането е проблем за намаляването на комплексността на даден граф. Най-оптималното редуциране на графа се отнася към клас от проблеми, наречен NP-сложност, за който все още не е открито унифицирано и ефикасно решение [19]. Ето защо, асемблаторите се обръщат към евристичните алгоритми и апроксимиращи решения, с цел да се избяга от повторяемостта, поправянето на грешки, намаляване на сложността, разширяване на опростените пътища, и в крайна сметка оптимизиране на графа.

5.4. Алчни граф²-базирани асемблатори

Първата група пакети за асемблиране на данни от NGS технологии използват т.нар. "алчни" алгоритми. Те са описани в обзорите [20], [21].

Този вид алгоритми изпълняват по една основна операция: при даден прочит или контиг, добавят още един прочит или контиг. Тази операция се повтаря докато добавянето престане да бъде възможно. Всяка операция използва следващия елемент с най-висока степен на припокриване със входния. Оценяващата функция измерва например броя на съвпадащите нуклеотидни бази в региона на припокриване. Ето защо контигите нарастват чрез "алчни" разширяване, винаги вземайки прочита, който е намерен, следвайки постановката за най-висока оценка за припокритието.

Това са имплицитни граф алгоритми. Те драстично опростяват самия граф чрез

² Greedy graph

приемането за единствено вярно решение само рамената с най-висока оценка. Възможна е оптимизация, като се приеме, че може да съществува едно единствено припокриване за всеки от краишата на прочита, който обработват. Друг вариант е, вече използваните препокривания да бъдат изключвани от по-нататъшната обработка на данните, след приключване на работата по контига.

Както всички асемблатори, тези алгоритми се нуждаят от механизми, чрез които да се избягва включването на False-Positive припокрития в контигите. Припокритията индуцирани от повтарящи се секвенции обикновено са с по-голяма оценка от колкото оригиналните. Асемблатор, който използва false-positive припокрития може да състави контиги, които включват несвързани секвенции, и по този начин да произведе химера.

Кратко описание на по-известните асемблатори:

SSAKE [22] беше първият създаден инструмент за асемблиране на къси прочити, проектиран за несдвоени къси прочити с еднаква дължина. Основава се на идеята, че високата степен на покритие води до “навързване” на прочити с нулево ниво на грешка, ако прочитите с отчетени грешки могат да бъдат избегнати. Инструментът не прилага алгоритмите за графи. Вместо това се разчита на таблица за късите прочити, индексирани по техния префикс. SSAKE претърсва итеративно таблицата за къси прочити, които се препокриват с единия от краишата на контига. Прочитът-кандидат трябва да притежава идентичен регион на припокритие prefix-to-suffix, чиято дължина е около поносимия праг. SSAKE избира внимателно между многото прочити с еднаква дължина на препокриващия регион. Първо се предпочитат прочити със end-to-end потвърждение в другите прочити. Второ - софтуерът засича кога дадена съвкупност от кандидати представлява множествени разширения. В основни линии се определя кога суфиксът на даден кандидат-прочит се различава и тази разлика е потвърдена в други прочити. Това е еквивалентно на намирането на клон в даден граф. Тук софтуерът приключва със съставянето на контига. Потребителят може избере дали да промени “строгите” правила, като в този случай SSAKE приема прочитът с най-висока оценка. Когато липсва прочит, който да удовлетворява първоначално зададения доверителен интервал, програмата продължава, като снижава критериите до достигане на втория минимум. Ето защо, от потребителя зависи колко агресивен ще бъде софтуера в намирането на решение, чрез регулиране на границите за регионите с ниско ниво на покритие. SSAKE беше разширен, така че да работи и с прочити със свързани краища, а така също и с неоптимизирани асемблирани прочити [23].

SHARCGS [24] също работи с прочити с фиксирана дължина, високо покритие, и несдвоени къси прочити. Той добавя пре- и пост-процесинг функционалност към алгоритъма на SSAKE. Препроцесингът включва филтриране на прочитите, съдържащи грешки чрез поставяне на изискване за минимален брой към пълната дължина, която точно препокрива другите прочити. Възможно е използването дори на “по-строг” филтър, който изисква комбинираната QVs на припокритие при прочитите да достига минималните отклонения от поставените граници за оценката. SHARCGS филтрира три пъти прочитите, като всеки път условията са различни. По този начин се създават три нови съвкупности от данни, които биват асемблиирани по отделно. В етапа на пост-процесинг контигите от трите асемблиирания биват сравнявани. Това смесване цели да увеличи дължината на контигите, изградени от групата данни с най-прецезни филтриране, чрез интегриране с тези от другите два по-ниска класа.

VCAKE [25] е още един итеративен допълващ алгоритъм. За разлика от своите предшественици, той може да включва неоптimalни припокрития, в процеса на създаването на контигите. VCAKE в последствие беше комбиниран с Newbler, като се създаде “pipeline” за хибридните данни, генеририани от платформата за секвениране Solexa+454 [26].

Друга поредица от комбинации на софтуер в една линия е тази между Newbler и Celera асемблатора за хибридните данни от 454+Sanger [27]. И двете схеми “споделят” контиги от първия асемблатор за да се произведат псевдо прочити, подходящи за втория асемблатор. По-късните съчетания на софтуери регулират покритието на прочитите и качеството на нуклеотидните бази в псевдо прочити, които се генерират. По този начин се подпомага втория асемблатор да даде подходящо тегло за контигите с високо покритие от първия асемблатор, например при съставянето на консенсусни бази.

5.5. Препокриващи/Подреждащи/Консенсусни асемблатори³

Подходът OLC е типичен за асемблаторите на данни, генериирани от по-старата технология Sanger. Той е оптимизиран за голяма група геноми в софтуера Celera Assembler [28], Arachne [29], [30], и CAP и PCAP [31]. Написани са обзорни статии за [32]–[34].

Асемблаторите от този тип използват препокриващ се граф. Операциите се развиват на три етапа:

1. Откриването на препокриващите се региони се извършва чрез множествено

³ Overlap/Layout/Consensus Assemblers (OLC)

сравняване на секвенциите, а така също и сравняване по двойки. Сийдащи и разширенни евристики са използвани, за да се подобри ефективността на алгоритъма. Софтуерът преизчислява K-мер-ите за всички прочити, като се избират кандидат - прочити, които споделят едни и същи K-мер-и. Между K-мер-ите на кандидатите, отговарящи на поставените условия се правят секвенционни сравнявания. Процедурата по откриване на препокриващи се региони е чувствителна към настройките за дължина на K-мер-а, минималната дължина на препокриваща се регион и минималния процент на идентичност, изискван за дефиниране на прекритие. Тези три параметъра засягат устойчивостта при грешки, в самите нуклеотидни бази и ниското покритие при секвенирането. Големите стойности на тези параметри водят до съставянето на по-къси контиги, но не и на по-точни и верни такива.

2. Конструирането и манипулацията на препокриващия се граф води до приблизителна подредба на прочитите. Препокриващите графи позволяват да не се включват секвенционните варианти на отделните бази, което е предпоставка големите геномни графи да могат да бъдат подложени на компютърна обработка без да предизвикват недостиг на RAM памет.

3. Множественото сравняване на секвенции (MCC) при асемблирането се характеризира с прецизност на подредбата на прочитите и съставянето на консенсусната верига. Основен недостатък на метода е, че липсва ефективен начин на изчисление на оптималния вариант (Wang and Jiang 1994). Ето защо, на фазата "съставяне на консенсусната секвенция" се използва прогресивно сравняване по двойки прочити, като основен критерий например може да бъде добрата приблизителна подредба на прочитите. На този етап, алгоритъмът изисква зареждане в паметта на компютъра на нуклеотидните бази. Процесът може да бъде. Най-широко използваният софтуер в тази област е Newbler (M. Margulies, et al. 2005), дистрибузиран от 454 Life.

Newbler: Първата версия е описана в притурка на сп. "Sciences", като софтуер, който получава за вход целеви несдвоени прочити, всеки с приблизителна дължина от 100 базови двойки, генериирани от GS 20 секвенатор. Софтуерът има последваща ревизия, която се състои главно в добавяне на възможността за съставяне на скелето от сдвоени прочити. Както е описано през 2005-та година, Newbler имплементира двойно прилагане на OLC - първата фаза генерира т. нар.

“unitigs” от прочитите. Те представляват мини-асемблиирани контиги, които в идеалния случай са непрепокриващи се с прочити от други такива [28]. Унитигите служат като предварително създадени, строго консервативни контиги, които се използват в последващото асемблиране. Втората фаза на използване на OLC генерира по-дълги контиги на база унитигите. Това асемблиране се осъществява чрез припокриване по двойки на унитигите. Възможно е един унитиг да бъде разпокъсан ако префикс или суфикс му съвпадат при сравнение с друг контиг. Разцепването на унитига може да разпокъса индивидуални прочити, като доведе до това че прочитите могат да принадлежат на множество контиги. Такива прочити могат да бъдат химери или извлечени от граница на повтори.

Софуерът Newbler използва покритието с цел да се преодолее грешката, дължаща се на отделните нуклеотидни бази. В основни линии, може да се каже, че се прилагат метрични инструменти за изчистването на грешки, породени от броя на базите в хомополимерните повтори. Пресмятат се унитигите и консенсусните контиги във “поточното пространство”, чрез предоставената от платформата сила на сигнала, свързан със всеки поток на сигнала за определен нуклеотид. Нормализираният сигнал е пропорционален на броя на директните повтори на всеки нуклеотид за определена позиция. Консенсусните изчисления за поредицата от нуклеотидни бази са еквивалентни на закръгляне на стойностите преди усредняването им, което води до намаляване на прецизността. За всяка колона в MSA⁴, Newbler изчислява средния сигнал и го закръгля целочислено, за да може да се състави консенсусната верига.

Този пакет предлага възможност за “de novo” асемблиране, и включва подробно ръководство за потребителя. Разпространява се със 454 секвенаторите. Постоянните обновления са част от него, а всяка нова версия пристига с кратко описание на промените и нововъденията в прилагания алгоритъм. Сорс кодът не се разпространява.

Celera Assembler[28]: е адаптиран OLC асемблатор към данни от 454 платформата на база Sanger [35]. Адаптираният алгоритъм е наречен CABOG, открива препокриващите се региони чрез използване на “compressed seeds”. CABOG намалява хомополимерните “рън”-ове, които представляват многократни повтори на единични бази, с цел намаляване на грешката в дължината на хомополимера. Изграждат се първоначални унитиги, като се изключват прочитите, които представляват подстрингове на други прочити. Пресмятат се поднизовете за голяма част от данните, като изходните секвенции

⁴ Multi Sequence Alignment – множествено сравняване на секвенции

от 454 технологията са с висока степен на покритие, т.к. последните съдържат голямо разнообразие от прочите с различна дължина. Алгоритъмът избягва поначало поднизовите прочити, защото те в голяма степен са отговорни за съставянето на грешни контиги, т.е. получава се фалшиво припокриване.

Алгоритъмът прилага корекция на отделните нуклеотидни бази за пръв път описана за Arachne [29]. Сравнява се всеки прочит спрямо множество от прочити, с които той има препокриващи се региони. Намалява грешката от секвениране за всяка “проблемна” база чрез избиране на най-доброто припокриване. Не се извършва корекция прочита, а по-скоро се настройват нивата на систематичната грешка в регионите на припокритие с ниски нива. След това, се прилагат нива на допустимата грешка, зададени от потребителя. От така филтрираните припокрития, отговарящи на нивата, зададени от потребителя, и изискването за минимална дължина на района на припокритие, алгоритъмът избира най-добрая вариант за всеки един от двата края на прочита. По дефиниция най-добрият трябва да съдържа възможно повече бази. Презумпцията на това филтриране е да се избегне прилагането на друг бавен алгоритъм, който е бил използван първоначално от Celera Assembler [13]. Конструира се граф, съдържащ както прочитите, така и най-добрите припокрития. Чрез него се съставят унитиги, на базата на максимално елементарни пътища, не съдържащи разклонения и прекъсвания. След това се съставя граф от унитигите и сдвоените прочити, използван за изграждането на контиги, а контигите в скеле. Прилагат се серия от редукции на графа, включително премахване на вероятни транзитивни краища. Последната стъпка е съставянето на консенсусната секвенция, чрез прилагане множествено сравняване на база полученото “скеле” и прочитите.

Съществуват два семблатора, които прилагат подхода на OLC върху къси прочити от платформите Solexa и SOLiD. Софтуерът Edena [36] изключва дублиращите се прочити и открива всички идеални, с нулева грешка припокрития. Алгоритъмът “изрязва” т.нар. върхове и балони. Софтуерът е създаден за обработка на несдвоени прочити с еднаква дължина. Софтуерът Shorty [37] разглежда частен случай, при който няколко дълги прочита се използват, за да бъдат създадени къси прочити и техните mate pairs. Процесът е итерационен, като контигите се използват за съставянето на супер контиги.

5.6. Приложения на алгоритми с Граф на Де Брюйн

Третият подход за асемблиране е граф на Де Брюйн и е най-широко използван при къси прочити от платформите Solexa и SOLiD. В основата му лежат К-тегните графи, които със своите атрибути го правят атрактивен при големи

количества къси прочити. Този тип графи не изискват предварително откриване на препокриващите се райони чрез изчисления от типа “всеки срещу всеки”, не се налага помненето на всеки прочит или неговите припокрития, и в същото време избягва секвенциите, не отговарящи на минималните критерии за качество. Дори напротив - K-мер графът съдържа истинските секвенции, а при големи геноми графът може да изчерпи наличната памет. Този подход често пъти предполага използването на разпределена памет.

Подходът K-мерен граф е прилаган за асемблиране на прочити от платформата Sanger [14], като се основава на алгоритми, разработени [38] за дори по-стари технологии за секвениране [20]. Подходът често пъти се описва като граф на Де Брюйн, който представлява Ойлеров граф [39]. Този подход се основава на идеалния сценарий, при който разполагаме с перфектни данни (чисти от грешки). Реалните данни създават редица казуси, които усложняват алгоритъма и използването на K-мер-и.

Конструирането на графа става по-бързо когато се използват т.нар. хеш таблици, в които се прави претърсване за всеки K-мер. Въпреки че хеш таблиците изискват и консумират допълнително памет, K-мер графът съхранява само по веднъж всеки един от K-мер-ите, независимо колко пъти K-мерът се среща в прочитите. От гледна точка на използвата компютърна памет, графът заема по-малко ресурси отколкото прочитите, като едновременно с това биват прочетени и K-мерите.

Софтуерният подход, описан от Pevzner [38] се занимава с проблемите, които генерират геномните повтори. Повторите предизвикват цикличност в графа. Това означава, че са възможни различни крайни решения - пътища и съответно асемблиране и консенсусни вериги. Авторите Idury and Waterman [14] също работят по проблемите с реални данни. Те добавят два допълнителни типа информация към графа, и наричат резултата секвенционен граф. Всяко рамо носи етикет с прочитите и позициите на K-мера във всеки един от прочитите, в които K-мерът е наличен. Когато върховете имат едно входящо рамо и едно изходящо, тогава трите елемента (върха и двете рамена) биват обединени в едно рамо. Този процес е наречен елиминиране на “самотните”. Ойлеровите графи се оказват много добре поставени при платформите на Illumina, които произвеждат много по-голямо количество данни с много къси прочити с еднаква дължина.

Съществуват четири основни фактора, които усложняват прилагането на K-мер графите за асемблиране на ДНК секвенции.

- 1) ДНК има двойна верига. Права секвенция, от който и да е прочит може да се припокрие както от права, така и от обратна секвенция от другите прочити. К-тег графът съдържа върхове и рамена както от правата, така и обратната секвенция. Това предполага създаването на допълнителни механизми, които да предотвратят цялостно двойно асемблиране [14]. Отделна имплементация съхранява правата и обратната секвенции, заедно като подобни полу-върхове, като се налага ограничението, че пътищата трябва да влизат и излизат от един и същ полу-връх [15]. Друго решение представя алтернативните вериги на ДНК секвенциите, като един връх с две страни, като тук пътят трябва задължително влезе от едната страна, а излезе от противоположната [40].
- 2) Реалните геноми представляват сложни повторяеми структури, включващи тандеми от прочити, прочити - включени в други прочити. Повторите, които са по-дълги от размера на K-тега, водят до мрежа от K-тегни графи, което усложнява задачата с асемблирането. Идеалните повтори с големина K или повече се свиват вътре в графа, като оставят впечатлението за локалния граф за въже с разплетени краища; пътищата конвергират с дължината на повтора и след това дивергират. Успешното асемблиране изисква разделяне на конвергиралите пътища, които представляват свит повтор.
- 3) Палиндромът е ДНК секвенция, която е сама на себе си обратно комплементарна. Полиндромите предизвикват създаването на циклични пътища, т.к. водят към тях самите. Съществува поне един асемблатор, който дава много елегантно решение на този проблем - Софтуерът Velvet [15], поставя изискване към K - дължината на K-тега, да бъде нечетно число.
- 4) Реалните данни винаги съдържат грешка от секвенирането. Асемблаторите от класа на Де Брюйн графите използват няколко техники, за да намалят чувствителността към този тип грешки. Първо, прочитите биват подлагани на препроцесинг, с цел изчистване на грешките. Второ, върховете на графа биват претеглени чрез броя на късите прочити, които се отнасят към всеки връх, след което слабо претеглените графи биват отстранени. Трето, те преобразуват пътищата в секвенции и използват алгоритми за сравняване на секвенциите, с цел обединяване (съкращаване) на сходните пътища. Много от тези техники идват от класа на ойлеровите асемблатори.

5.6.1. Графът на Де Брюйн в Ойлеровите имплементации

Съществуват разработки на софтуер на базата на Ойлерови графи за прочити произведени от платформата Sanger [16], [39], [41]. По-късно имплементациите са модифицирани, така че да са приложими спрямо секвенциите за платформите 454 GS20 [42], и при технологиите за секвениране, генериращи както несдвоените къси прочити (Illumina/Solexa [43]), така и сдвоените прочити (Solexa [44]).

Прочитите биват първо филтрирани и след това се създава граф на Де Брюйн. Филтърът засича грешните бази в секвенциите чрез анотиране на ниско честотните K-теги. Критериите на тази предварителна стъпка зависят от: 1) късите прочити-повтори: повечето K-теги трябва да могат да бъдат открити в няколко секвенции; 2) случайния характер на грешката от секвенирането: за всяко K, където $4K$ превишава два пъти дължината на генома, повечето грешни K-теги трябва да бъдат уникални. Имплементацията включва списък на K-тегите и техните честоти на срещане в късите прочити, като биват изключвани или коригирани тези с ниска честота. Корекцията е особено важна за секвенциите с голямо ниво на грешка и голямо покритие. Тя редуцира общия брой на K-тегите, и по този начин броя на възлите в графа също. Рискът, който се поема е един истински полиморфизъм да бъде отчетен като грешка и да бъде поправен или елиминиран [39]. Съществува възможност за валиден K-mer с ниско покритие да бъде отстранен. Има случаи, в които прочитът може да бъде счетен за неверен, ако е прието че в него може да се наблюдават няколко различни K-мера, но никога заедно. Асемблаторите от тип OLC имат аналогична стъпка за корекция на грешните бази, която използва регионите на припокритие вместо K-тегите.

Процесът на Ойлерово филтриране се нарича още “спектрално сравняване” [39]. Софтуерът идентифицира грешките от секвениране чрез сравняване на K-тегите между отделните прочити и всички прочити. Приемат се за неверни индивидуалните K-теги, чиято честота във всички прочити е под доверителния праг. Последният се определя след като се види какво е разпределението на K-mer честотите в секвенциите. Обикновено разпределението е би-модално, като първият пик представлява K-тегите, които се срещат един или два пъти вследствие на грешката от секвениране (или на ниско покритие). Вторият пик представлява K-тегите, които се повтарят. Те могат да бъдат индуцирани както от покритието на прочитите, така и от повторите в генома. Избира се доверителен интервал между двата пика, като K-тегите се разделят на “добри” и “лоши”. Следва процедура за тестване на всеки прочит. За всяка секвенция с лоши K-теги, се прилага “greedy” изследване за субституция на базите, които

биха намалили броя на “лошите” K-мери [44]. Финалната стъпка предполага или приемане на напълно коригирания прочит или отхвърлянето му, като елиминираните секвенции могат да бъдат присъединени след асемблирането. Следва да се отбележи, че по този начин се филтрират грешките от субституцията, но не и инсерциите и делециите в символните низове на генома. Субституциите на базите са най-честия вид грешки при данните от платформата Solexa [45]. Създава K-мер-ен граф от филтрираните и коригирани прочити. След това се прилагат серия от манипулации върху графа, с цел да се елиминират ефектите от грешките при секвениране и геномни повтори.

Конструирането на K-мерния граф игнорира дългите последователности в прочитите. Ойлеровата имплементация внася корекция на този проблем чрез обработването на прочитите в графа на принципа на нишките при многозадачните системи за обработка на данни, което създава явна предпоставка за паралелизация на процеса. Картрирането на даден прочит върху графа става лесно. Поне първоначално, K-мерите в прочитите се картират спрямо уникални възли и прочитите отново са последователни и в съответствие с някой от пътищата в графа. Използването на картографирането е по-сложено. Края на прочита в един повтор са съвместими с всеки път, който съдържа този повтор, но останалата част от повтора е съвместима само с определен кръг от пътища, чийто брой е значително по-малък. Освен това, процесът на обработване на прочитите по тази нишкова технология, изважда едно парче от стринга от усукания като въже шаблон, като по този начин се получава едно копие на повтора. Така, нишковото обработване на прочитите генерира група от валидни пътища в графа. Това е възможно само когато дължината на повтора е в рамките между дължината на K-мера и дължината на прочита.

Сложният граф може да съдържа множество варианти на пътя между два връха, които съответстват на обратните краища от една двойка сдвоени прочити. Всеки един от възможните пътища представя един от предполагаемите варианти на ДНК секвенцията. В много случаи, само един от всички тези пътища генерира секвенция с необходимата дължина, отговаряща на ограниченията на сдвоените прочити. Ойлеровият граф осигурява свиване на това пространство за претърсване, като използва ограниченията, поставени от сдвоените прочити под формата на граници, в които трябва да се движи общата дължина на секвенцията - продукт на свързването на двета прочита, през които преминава избрания път. Имплементация идентифицира рамената, които се повтарят и ги премахва от множеството генериирани пътища. Това е еквивалентно на разсичането на контиги при границите на повторите в OLC

асемблирането.

В много платформи, късите прочити съдържат ниско качествени сигнали за определяне на нуклеотидните бази в 3' краищата на секвенциите. Решението, използване на Ойлеровите графи е даването на предимство при четене на префиксите, пред четенето на суфиксите. Избират се достоверни префикси още по време на стъпката за корекция на грешката от секвенирането. Дължината на префикса може да варира за всеки прочит. По време на процеса на поточна обратка на прочитите, префиксите и суфиксите могат да бъдат картирани спрямо множество пътища. Използването на евристични методи, позволява на Ойлеровата имплементация включва избор на картирианията, които са значително по-добри в сравнение с втория възможен най-добър избор. Суфиксите добавят покритие към множественото секвенционно сравнение, като по този начин те добавят нови връзки към графа. Всяка допълнителна секвенция води до увеличаване дължината на контига. Оставя се непроменено асемблирането на консенсусната секвенция, базирана на суфиксите, така че картираните суфикс допринасят единствено за осигуряването на свързаност.

Препокриващите се пътища са чувствителни спрямо дължината на минималния праг, а K-mer графиките от своя страна са чувствителни спрямо параметъра K. Високите стойности на K разрешават наличието на по-дълги повтори, но в същото време нарушават асемблирането в районите с ниско покритие на прочитите. Преодолява се този проблем отново, използвайки евристични методи. Конструират се и се опростяват два K-mer-ни графа с различни стойности за K. Идентифицират се дъгите, представени в графа с по-малка стойност за K, които липсват в графа с по-голяма стойност за K, и се определят като псевдо рамена във втория граф. Така се получава увеличаване на дълбочината на покритие, и съответно увеличаване дължината на контигите при асемблиране. Тази техника е аналог на подхода за запълване на “интервали” при OLC асемблаторите [28].

Някои от софтуерите, базирани на Ойлерови графи използват различна структура, наречена А-Брюйн граф. Наименованието идва от комбинацията на Де Брюйн граф и матрица на съседните върхове (*adjacent endpoints*), наречена А-матрица. Върховете в графа представляват последователни колони в множественото секвенционно сравнение. В сравнение с върховете в K-mer-ния граф, които представляват отделни прочити, съседните върхове са по-малко чувствителни спрямо грешката от секвенирането. А-Брюйн графът се използва за конвертиране на геномната секвенция в граф от прочити и класифицирани повтори. Тази схема е предложена като вариант за асемблиране от [16].

5.6.2. Асемблатор Velvet

Velvet [15], [46] е проверен, надежден и лесен за използване софтуер, който в основата си представлява DBG асемблер. Залага се на симплификацията на графа, за да се намали броя на простите непресичащи се пътища към единични върхове. Опростяването компресира графа без да се губи информация. Тази стъпка се прилага още при самия етап на конструирането на графа, като се провежда и още няколко пъти по време на процеса на семблиране. Техниката е аналогична на елиминирането на singletons при K-тегните графи [14], и на формирането на унитиги при препокриващите се графи [15] и на OLC асемблаторите [28].

Почистването на K-тег-ния граф става чрез итеративно премахване на разклоненията. Алгоритъмът е подобен на Ойлеровата процедура. Елиминирането на разклоненията води до драстично намаляване на размера на графа при реални данни [15]. Това се дължи най-вероятно на факта, че преди всичко се прилага филтриране на грешките в базовите позиции. Тук не се прилага Ойлеровият спектрален сравняващ филтър. Използва се параметър за минимален брой повторения в прочитите за всеки K-тег, за да бъде даден прочит удобрен за връх в графа.

Софтуерът Velvet намалява комплексността на графа като се ограничава в търсене на т. нар. “балони” в графа. Алгоритъмът условно е наречен “туристическа обиколка” (tour bus algorithm). Той използва претърсване по широчина, ветрилообразно, със широчина - широчината на образуваното ветрило от рамена, започвайки от върховете с множество изходящи дъги. Подробното претърсване тук не е подходящо, т.к. всеки балон може да съдържа други балони. Затова търсенето е ограничено, като по този начин се осигурява податливост на обработка на самия граф. Кандидат-пътищата се обхождат итеративно за всеки връх и всички пътища, докато дължината на пътя не достигне необходимия тресходд. Търсенето на кандидат-балоните се свежда до тези, които имат секвенционно сходство в алтернативните пътища. Когато бъде открита балонна структура, софтуерът премахва пътя, генериран на базата на няколко прочита и вън от графа се прави повторно сравнение на прочитите, започвайки от премахнатия до края на съответния път. Тази операция може да доведе до ефекта “papering over” по отношение на истинските различия в секвенциите, което се дължи на полиморфизма в донорната ДНК или на прекаленото свиване на почти идентични повтори. Алгоритъмът, който използва Velvet е подобен на елиминирането на изпъкналости при Ойлеровите графи [16], и е аналогично на детекцията и “изглажддането” на балоните при OLC асемблаторите [18].

Следва намаляване на комплексността на графа чрез поточно обхождане на прочитите. Премахват се пътищата, които обхождат от по-малко количество прочити от колкото е заложено в тресхода. При тази операция има риск да бъдат елиминирани секвенции с ниско покритие, но се счита тя премахва предимно пътища, в които участват прочити със секженциона грешка. Когато се разполага с дълги прочити, спрямо тях се прилага алгоритъм, наречен “Rock Band”. Създават се върхове вън от графа, които получават подтвърждение, че са удачни от два или повече дълги прочита.

Последното съкъщаване на графа включва използването на т.нар. “приятелски двойки”. Ранните имплементации използват алгоритъм, наречен “трохи” [15], който е подобен на “приятелските двойки” при DBG алгоритмите [41] и на запълването на “дупките” при OLC алгоритмите [28]. Той обработва двойки от дълги контиги (семпла пътища), свързани чрез сдвоени краища. Дългите контиги се използват като котви, разстоянието между които се запълва с къси контиги. Събират се всички къси контиги, имащи връзка с дългите контиги, и върху тях се прилага претърсване по широчина чрез DBG за свързване на дългите контиги и чрез обхождане на късите контиги. По-късните версии на софтуера използват алгоритъм, наречен “морско камъче” [46]. В този алгоритъм уникалните и повтарящите се контиги се заменят съответно с дългите и къси контиги, продукт на алгоритъма на “трохите”. Класификатора за уникалност/повторяемост е базиран на покритието на прочита в контига. Използва се статистически тест по подобие на асемблатора Celera - A-stat [28], като се сравнява с дадено очаквано покритие. Изполва се даденото (за всяка библиотека) разпределение за дължината на инсерцията, за да се получи прилизителния вид на контига. Прави се претърсване на DBG, с цел откриване на път, който отговаря на прилизителния вид на контига.

Има възможност Velvet да бъде пуснат няколко пъти за един същ пакет от данни, с цел да се оптимизира избора на три критични параметъра. Дължината на K-мерите трябва да бъде нечетен, за да бъде изключена възможността за наличие на палиндромни повтори. Очакваната минимална дължина на K-мерите в прочитите определя кои K-мери априорно няма да се използват. Очакваното покритие на генома в прочитите контролира прекъсването на връзката.

В заключение може да се каже, че Velvet предлага цялостна имплементация на асемблиране от тип DBG. Не се използва препроцесинг на данните с цел отстраняване на грешките, въпреки че съществува четящ филтър за избягване на грешките. Прилагат се серия от евристични процедури, които намаляват

сложносста на графа. Евристиките използват топологията на локалния граф, покритието на прочитите, идентичността на секвенцията и сдвоените краища като форми на ограничения. Целта на софтуера е да се направи асемблиране на де ново секвенирани прочити със сдвоени краища, които са продукт на платформата Solexa. Има добавка, която позволява асемблиране на данни от платформата SOLiD [47]. Към момента, не е възможно използването на Velvet за много големи геноми, т.к. софтуерът изисква голямо количество операционна памет, което се явява и основният му недостатък.

5.6.3. Асемблатор ABySS

Софтуерът ABySS е имплементация [40], проектирана с оглед използването на ограничени компютърни ресурси за асемблиране на геноми с размери, подобни на тези от групата на млекопитаещите организми, като се използват алгоритмичните подходи на DBG. В основата на софтуера е залегната технологията за грид компютърните системи, при които се извършва разпределение на задачите, и така всеки компютър (респективно разполагаемата операционна памет) обработва даден K-теген граф и извършва част от изчисленията по графа. По този начин успешно се асемблират 3,5 мил. прочита, генериирани от платформата Solexa от човешка ДНК.

Съществуват няколко проблема при осъществяването на такава грид система за асемблиране, свързани с алгоритъма за обработка на данните от един процесор. Трябва да се осигури възможност за паралелизация на целия алгоритъм, така че процесите да могат да бъдат изпращани към отделните машини в грида, а в последствие резултатите да бъдат обединени. Софтуерът разделя асемблирането въз основа на различните възли в графа, като всеки връх се обработва отделно, като самостоятелен процес, и към всеки процесор се пускат по няколко възела от графа, т.е. по няколко процеса едновременно. Това става като се извършва конвертиране на K-тегите в целочислено число. Формулата не зависи от това от кой край бива прочетен K-тега, така че обратната комплементарна верига има същата стойност. Тя би била полезна ако съседно картираните K-теги могат да бъдат пуснати за обработка към един и същ процесор. Не е ясно до каква степен това е постигнато.

Имплементацията на ABySS въвежда паралелни изчисления за намаляване на комплексността на графа, като итеративно се премахват разклоненията, които са по-къси по дължина от заложения праг. Изглаждането на балонните структури се постига чрез ограничено претърсване, като се предпочитат пътища, включващи повече прочити. Опростените пътища се използват за създаването на контиги и “усукване на нишките (пътищата)”. Използва се

компактно представяне на K-мерния граф. Всеки връх в графа представлява K-mer, и неговата обратно-комplementарна верига. Възелът съдържа допълнителни 8 бита информация: наличие/отсъствие на една четирите бази в края на секвенцията като допълнение. Използват се алгоритми за паралелен обход на графа, разпределяйки ги между процесорните ядра. Преходът от един възел към друг се извършва много елегатно: формира се цифрово адресиране, което включва последната база от K-мера, плюс разширението от една база в края, което указва реброто, както и процесора, който обработва следващия възел от графа. Когато пътят минава през връх, който се обработва от друг процесор, се изльчва заявка за комуникация към съответния процесор, обработващ върха. Комуникацията между отделните процесори обикновено е бавна и затова докато се изчаква отговора (т.е. осъществяване на връзката), стартират други процеси за останалите върхове в графа.

След тази паралелизация се използва техниката на сдвоените краища, за да бъдат съставени контигите, като не се създават скелети на сдвоените краища. Може да се обобщи, че софтуерната имплементация ABYSS е подходящ софтуер за асемблиране на къси прочити, генериирани от платформата Solexa и на прочити със сдвоени краища.

5.6.4. Асемблатор AllPaths

Софтуерът AllPaths е асемблатор от тип DBG, предназначен за обработка на големи геноми. Първоначално е публикуван с резултати, получени от симулирани данни [48] а по-късно излиза с ревизия за реални данни [49].

Използва се препроцесинг на данните, свързан с корекция на прочитите на база спектрално сравняване чрез Ойлеров подход. Приемат се за правилни K-мер-и, които се срещат с висока честота в прочитите и високо качество, където всяка база от секвенцията трябва да бъде подтвърдена от минимален брой сигнали за определяне на нуклеотидната база, с коефициент за QV по-голямо от определения праг. Филтрирането на K-мерите се извършва за три различни стойности на K-мера, като ефективността се постига в две насоки. Прочитите биват възстановени ако две или повече замествания с ниско качество на QV на отделните нуклеотидни бази постигат доверен K-mer, или K-мерите се възстановяват по-късно, ако те са от съществено значение за изграждането на пътя между двойките прочити със свързани краища.

След това се извиква втора процедура по препроцесинг, която създава т.нар.“unipaths.” Процесът започва с изчисляването на идеалните припокрития на прочитите, базирани на K-мерите, като на всеки “navързан” K-mer се

присвоява идентификационен номер. Генерира се база от данни с идентификатори на интервалите и на връзките между прочитите. Интервалите се вмъкват, така че да отговарят на всички прочити. Операцията е еквивалентна на създаването на DBG, последвано от елиминиране на singleton-ните. Имплементацията на базата от данни вероятно намалява изискуемата RAM памет, която е необходима за създаването на графа, което е следващата стъпка.

Създаване на DBG, въз основа на базата от данни. Първата стъпка тук е намаляване броя на разклоненията по графа, което е образно наречено “орязване на дървото”. Следва разделяне на графа, чиято цел е да реши проблема с повторите в генома, чрез асемблиране на регионите, при които липсва локална повторяемост. Прилага се евристика за да бъдат избрани частите от графа, които образуват непрекъснат път в генома. Резултатът е: 1. Набор от частични пътища с върхове, коресподиращи на дълги, с умерено покритие, широко раздалечени контиги; 2. Набор от частични пътища с върхове и прочити, свързани чрез сравняване и сдвоени краища. Интервалите между прочитите със сдвоени краища се запълват, чрез претърсване на K-тегните графи, в които съществува един единствен път, удовлетворяващ ограничението, поставено от дължината на интервала. За по-строги ограничения на вариацията, първо се използва по-малък брой от прочити със сдвоени краища. Всеки частичен път се асемблира по отделно и паралелно. След това се локалните графи се слепват, ако има райони на припокритие между тях. Този процес е аналогичен на слепването на контиги, въз основа на регионите на припокритие между секвенциите. В глобалния граф се използва евристика, за да бъдат премахнати разклоненията, малките несвързани компоненти и пътищата, които не са разделени от прочитите със сдвоени краища. Технологията със сдвоените краища се използва още и когато броят на повторите е намален, което си личи от модела на “протриване на оплетката на въжето”.

Като заключение, може да се каже, че обект на AllPaths са големите геноми за асемблиране, като се използва техниката на сдвоени краища на прочити, генериирани от платформата Solexa. Прочитите биват филтрирани на база качество, с цел да бъдат поправени грешки, резултат от заместването. К-тегният граф е опростен, като се изгражда въз основа на прочитите и препокриващите се региони. За генериране на графа се използва техниката “разделяй и владей”, при която глобалният граф е разделен на локални графи.интервал

5.6.5. Асемблатор SOAPdenovo

Софтуерът SOAPdenovo [50] генерира геномни секвенции за род

млекопитаещи, с високо качество, на базата на милиони къси прочити, секвенирани с платформата Solexa, чиято средна дължина е 75 базови двойки и дори по-малко [51]–[53]. Софтуерът е със свободно разпространение, но без достъп до сорс кода.

Асемблаторът SOAP филтрира и коригира прочитите, като използва *предварително оценени прагове* за K-мер честотите. Прочитите биват съединявани (асемблирани), а пътищата, които показват симетрично разплитане - разделяни. Премахват се “балоните” чрез алгоритъм подобен на Velvet, но с по-високо покритие на прочитите, определящи “оцеляващия” път. Асемблаторът представлява DBG имплементация, която се заимства от Ойлеровите графи и Velvet. Ето защо неговият граф е пространствено по-ефективен. Графът не използва проследяване на прочитите, и поради това изисква само 120 ГБ RAM памет за съхранение на 5 милиарда възела от 3,3 млрд. прочита (след филтрацията). Както е описано в [50], графът е конструиран така, че не се използват данни от големи библиотеки, защото те предполагат генерирането на много химерни прочити.

Асемблаторът създава контигите от прочитите чрез прилагане на DBG метод. де Брюйн графа бива съкратен чрез създаване на т. нар. “гръбнак”. Картират се всички свързани прочити спрямо консенсусните контиги, като се включват и прочитите, които DBG не е използвал. Създава се граф на базата на контигите, чийто рамена представляват двойки вътрешни контиги, поставящи ограниченията. Софтуерът намалява комплексността на графа от контиги, чрез премахване на връзките, които транзитивно произтичат от други такива. Също така се елиминират повтарящите се контигите и несъвместимите пътища, произтичащи от прочитите-повтори. Подобни техники са имплементирани и в софтуера CABOG, както и в неговия предшественик - Celera Assembler. SOAP използва двойките, за да прикрепи прочитите към интервалите, образувани от два съседни контига в един scaffold. Това е подобно на технологията, използвана от CABOG - “rocks and stones” [28], [53] и на Velvet - “breadcrumbs and pebble”. Асемблаторът използва Де Брюйн граф, за да извърши процедурата по асемблиране на прочитите, присвоени към всяка “интервал”. В обобщение може да се каже, че SOAP е софтуер, който обработва големи геноми и представлява особена комбинация от OLC и DBG техники, базиран на предходни NGS асемблатори.

6. Методи на размитите множества

6.1. Софт Компютинг и Биоинформатика

Геномните изследвания генерираят голямо количество данни, което се нуждае от методи на софт-компютинг за обработка[54] като изкуствени невронни мрежи, системи, използващи размита логика; еволюционни алгоритми, метаевристични и алгоритми, базирани на интелекта на рояка; статистически модели и алгоритми и т.н., които могат да се справят с обема на генерираната информация. Софт-компютинг [55], [56] методологиите поддържат както интелигентен процесинг, така и обработка на проблеми и задачи от реални живот. Те са толерантни към неточността, несигурността, апроксимацията и частичната достоверност на данните[57]. Предлагат проследяемост, бързина и нисък разход на ресурси чрез взимане на решения, наподобяващи човешката логика. Развитието на технологиите и науката доведоха до бързо увеличаване и преход от ниско-обемни данни към високо-обемни, не само в областта на ДНК секвенирането, но и в други области на биологията [58]. Това обуславя все по-широкото прилагане на методите на софт-компютинг във все по-разнообразните задачи, които поставя биоинформатиката. Развитието на методологията в областта на SC демонстрира повишаване на технологичните стандарти за алгоритмизация и апликации в биоинформатиката особено в области, свързани с паралелното геномно секвениране, сравняването на геномни фрагменти, претърсването на геномни бази данни, автоматизирането на геномното анотиране, моделиране и съхранение на хетерогенни бази данни и др. Проблематиката, която се поставя от биоинформатиката може да бъде най-общо обобщена като задачи в следните области за анализ на биологични данни: геномика, протеомика, микроареи, системна биология, еволюция и текст майнинг и общи за всички изброени [59].

Анализът на данни от геномиката и протеомиката обхваща най-важните и основни разработващи се инструменти, т.к. голямата цел е откриването на факторите за различните болести при човека и откриване на начини за въздействието върху тях[60].

Задачи на геномиката: геномно асемблиране, откриване на вариации в генома, картиране, откриване на гени (анотиране), предсказване на функции на гените, откриване и предсказване на генно регулаторни елементи и мрежи, анализ на некодиращи ДНК фрагменти.

Задачи на Протеомиката: предсказване на структурата и функциите на протеините, откриване на нови лекарства

Общи задачи: сравняване на ДНК/РНК фрагменти, откриване и корекция на грешки във фрагментите;

Задачи на микроареите: разпознаване на шаблони, класификационни проблеми [59].

Задачи на системната биология: откриване, дефиниране и симулиране на метаболитни пътища и процеси, анализ на сигналите подавани от различните клетки [61].

Задачи на еволюцията: построяване на филогенетични дървета за откриване на наследствените пътища, откриване на подобност и сходност между различните фрагменти или организми.

Задачи на текст майнинг-а: приема се за страничен ефект, създават се различни онтологии на база хетерогенна информация, както от литературни източници, така и от ДНК/РНК данни. Намира приложение при откриване на функционалности, клетъчна локация, протеинови взаимодействия [62].

Може да се каже, че биоинформатиката съчетава стратегии по генериране, съхраняване, организиране, архивиране, анализ и визуализация на данните.

Методите на SC успешно решават проблемите на биоинформатиката чрез селекция, кълстерилизация, обработка на сигналите, анализ на образи. Поради спецификата на SC методите, те са в състояние да се справят с многомерността и разнородността на данните, генериирани от биологията[63].

Теоретичните основи на SC методите ги поставят като най-близки и естествени, наподобяващи биологически процеси сами по себе си, за разлика от традиционните формални логически системи, и още повече – те са взаимно-допълняеми [54].

Таблица 1: Съотносимост между биоинформатичните задачи и методите на софт-компютинг, които се прилагат за тяхното решение

Задачи	Размита логика ⁵	Рояк & Мета евристични методи ⁶	Графи ⁷	Стат-ки методи ⁸
Сравняване	FL [64]	GA [7], [65]–[80] ACO [70], [81]–[86] EP [87] PSO [88]–[92] SA [93]–[96]	DG [97] SO [88]	DP [90] HMM [88], [98]
SNP детекция	ANN [99]–[107] BN [108], [109] SVM [110], [111]	GA [100], [105], [111]–[114] PSO [110], [114], [115] SI [116][117]	[86]	HClust [86] PCA [118] K-NN [113], [119] ICA [120]
Протеомика	ANN [121]–[126] ANN & SVM [127] FL [78], [80], [124], [128]–[135]	GA [136], [137] SI [138]–[140]		
Геномика	ANN [141]–[143] FL [131], [141], [144]–[146]	GA [147]–[156]		
Еволюция	FL [157]–[160]	GA [155]		

⁵ Fuzzy Logic, ANN, BP, SVM, Bayes Networks(BN)

⁶ ACO, PSO, GA, ABC, EP, Simulated Annealing (SA), Swarm Intelligence(SI)

⁷ De Bruijn Graph, Dispertion Grapf, Stochastic Optimization

⁸ Bayes Networks, HMM, DP, PCA, K-NN, H-Clusterization, ICA

6.2. Размити множества (Fuzzy Sets)

Методът на размитата логика се използва за обработка и анализ на данни, които притежават субективни оценки. Чрез създаването на размити мрежи от данни, имаме възможност освен характеристиките Да/Не, Истина/Лъжа и други бинарни оценки, да развием собствени мрежи от оценки или да дообогатим съществуващи, така че в крайна сметка да разполагаме с необходимите оценки. По този начин размитата логика позволява субективните лингвистични оценки да бъдат използвани в оценката и отчитането на неточността на информацията, с която се разполага, и която трябва да бъде анализирана. Размитата логика представлява допълнение към класическата предикатна логика, която се основава на бинарния принцип за истината. Тази теория най-добре се илюстрира от следните понятия:

- “Степен на принадлежност” (partial matching), което служи за обозначаване на степента на съответствие между дадена стойност на лингвистичната променлива и субективното значение влагано в нея. Този параметър приема стойности от 0 до 1, като единицата означава пълна убеденост в твърдението. Подходящ пример за онагледяване е лингвистичната променлива “ръст”. Ако приемем, че променливата има три значения: “нисък”, “среден” и “висок” в рамките от 1,5м до 2,3м., то можем да кажем, че ръст от 1,80м се счита за висок със степен на принадлежност 0,75.
- F.G-Generalizaton - означава че, всяка теория, техника, метод или проблем може да бъде представен чрез размито множество (f-generalized) чрез замяна на множеството с точни характеристики със множество с обобщаващи характеристики, т.е. с размито множество. Това обикновено става като се използва лингвистична скала. От тук следва и изводът за granulated или g-generalization. Т.е. всяка теория, техника, метод или проблем може да бъде разделен на подмножества, чрез образуването на информационни кълстери.
- ”Лингвистична скала”[169] - представлява последователност от размити характеристики на един и същ показател. Особеното при лингвистичните скали е че интервалите имат числови значения. Една скала е качествено построена когато интервалите следват плътно един след друг без да се препокриват.

6.3. Изкуствени Невронни мрежи (ИНМ)

6.3.1. Същност

ИНМ са компютърен аналог на биологичните невронни мрежи[170].

Неврон - ИНМ е изградена от множество неврони, които класифицираме като входни, изходни и междуинни (скрити), в зависимост от характера на информацията, която представляват.

Аксон - предават обработената информация, която е постъпила в неврона на следващия неврон, т.е. осигурява изхода от неврона. Всеки неврон разполага само с един аксон.

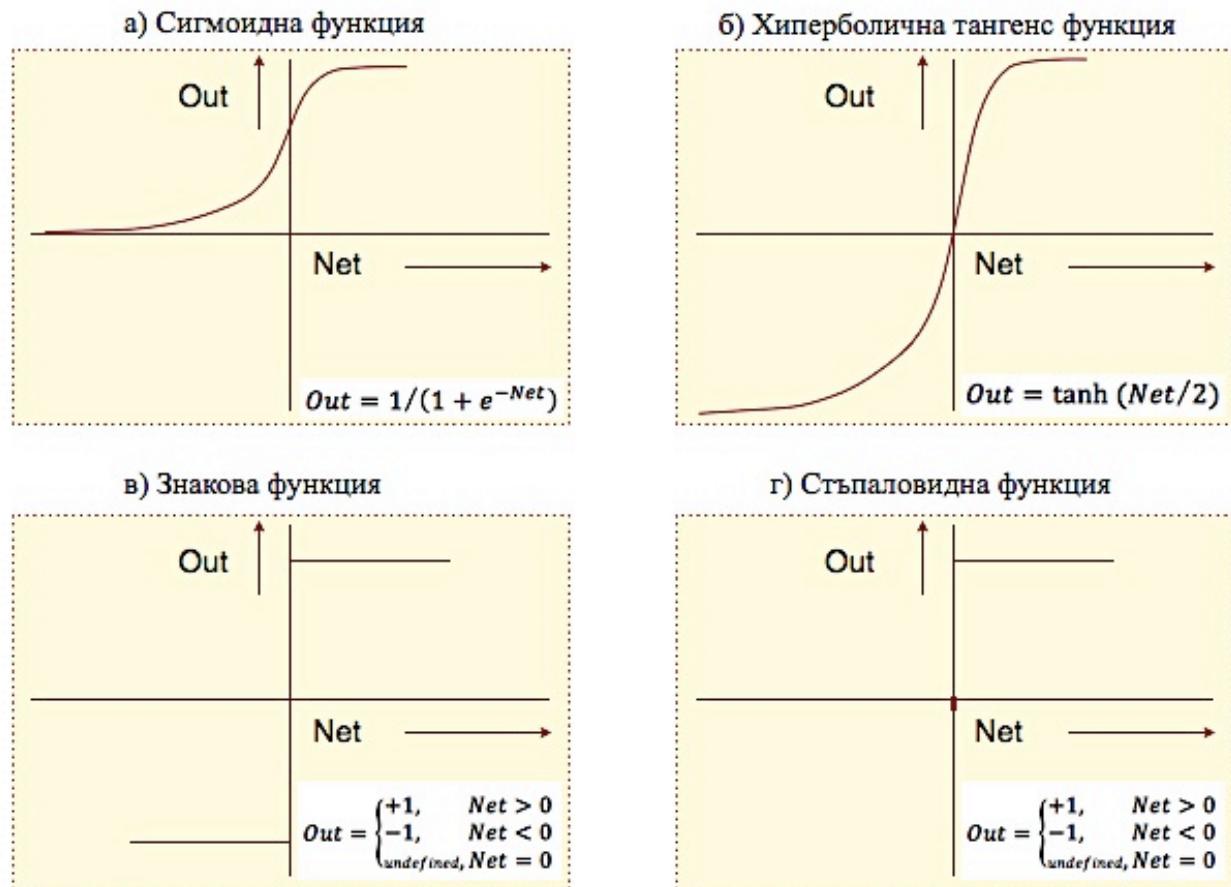
Дендрит - чрез тях сигналите от другите неврони постъпват в неврона като информация. Може да символизират както информация от външната среда, така и информация постъпваща от друг неврон. Всеки неврон разполага с множество дендрити. По тази причина входните сигнали от тях се сумират. Физиологически това довежда до повишаване или понижаване на електрическия потенциал на неврона. При достигане на определени нива, невронът изпраща точно определен сигнал, т.е. с определена сила и продължителност. От гледна точка на компютърната имплементация на този процес, тук имаме функция, която следи за тези прагове, като съответно подава и подходящо значение на функцията. В литературата я наричат активационна функция. Тя има линеен характер.

Синапс - точката на съединение между аксона на един неврон и дендрит на друг неврон. Прието е, че синапсът може или да активира или да подтисне възприемащия неврон, но никога и двете. Имплементацията на синапса се моделира чрез нелинейна активационна функция, която се нарича OUT. Най-често използваните нелинейни функции са:

- Soft нелинейни функции: използват се предимно при задачи за прогнозиране на бъдещи стойности
 - Сигмоидна функция: параметрите на този вид функция се уставяват експериментално.
 - Хиперболична тангента
- Hard нелинейни функции: използват се предимно при класификационни задачи
 - Сигнум функция
 - Прагова функция: невронът остава неактивен, докато претегленият вход net не достигне определена прагова стойност (T)

Независимо от типа на тази функция, нейните значения са заключени в интервала $[0,1]$ или $[-1,1]$. Тази амплитуда на сигнала се използва като един вид мащабиране на информацията.

Фигура 7: Видове функции



Отклонение (bias) - служи като входен параметър, който има ролята да поддържа определено ниво на активационната функция. Този елемент не е задължителен за всички невронни мрежи.

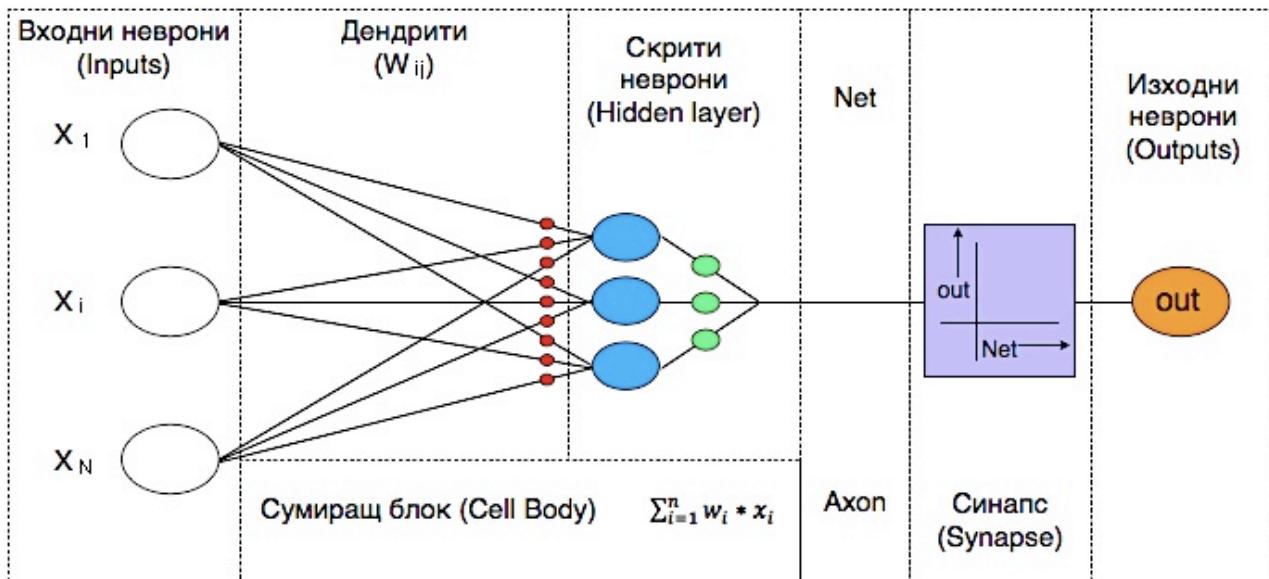
ИНМ наподобяват функционалността на биологичните мрежи по редица признаки:

- Натрупаната информация преминава през “курс на обучение”
- От обучението се натрупват знания за връзките между отделните информационни елементи. От информатична гледна точка това синаптичните тегла.
- НМ изпълняват разпределена обработка на информацията, като отделните изчислителни процеси и големият брой синапси подсигуряват добра производителност.
- НМ са много устойчиви от архитектурна гледна точка. Това е така дори при биологичните системи. Известно е че до едно и също решение може

да се стигне по различни пътища (архитектура), като не винаги това трябва да означава голяма разлика в разхода на време.

Силата на НМ е във възможността им адекватно да обработват всяка информация, включително неточна и непълна.

Изкуственият неврон има следния общ вид:



Фигура 8: Общ вид на изкуствения неврон

Тук **Cell Body** е сумирацият блок или т.нр. **net функция**. В нея се сумират произведенията от теглата и входните сигнали. Поради самия си характер функцията предполага, че колкото броят на невроните е по-голям, толкова по-малко влияние ще оказват грешките, т.е. шумът няма да оказва голямо значение.

6.3.2. Архитектура (Топология) на ИНМ

Външният вид на всяка ИНМ може да бъде представен в таблична форма, където всеки ред задава определен брой n входни параметри, и определен брой параметри m, които представляват информация за изхода. Колкото по-голяма (по-дълга) е таблицата, толкова по-лесно ще бъдат извлечени зависимостите между входните данни и резултатите. В процеса на извличането на данните и претворяване им в прогнози или класификация важно значение има топологията на мрежата. Т.е. таблицата, която виждаме далеч не е всичко. Бихме могли да разполагаме с x на брой групи от по y_i броя неврона, където $i=1..x$ и $y_i \Leftrightarrow y_{i+1}$.

В зависимост от задачата топологиите могат да бъдат по-прости и по-сложни. При всички положения трябва да се спазват следните правила при избор на архитектура на мрежата:

- Тестване на данните с възможно най-елементарните топологии, за да се придобие ориентация за данните - от една страна, и от друга - в определени задачи архитектурата е елементарна и не е необходимо да бъде усложнявана.
- Добра стратегия е новото междинно ниво да бъде брой неврони на половина от предходното.
- След експериментално установяване на ориентировъчната архитектура, да се изгради алгоритъм, който да тества с производни архитектури на ориентировъчната. За тези цели много добри резултати дава апарат на Генетичния алгоритъм.

Топологиите могат да бъдат разделени най-общо на прави и рекурентни. Казваме че една архитектура е с прави връзки, когато връзките са в посока Входни неврони, Скрити неврони, Изходни неврони. Когато се наблюдава някакъв различен вид насоченост говорим за рекурентни топологии:

- Еднослойна рекурентна със *feedback* структура (обратни, рекурентни) - тук се наблюдава връзка нeron от ниво $i+1$ към ниво i , както връзка насочена от неврона към самия него, т.e. това т.нар. bias или коригиращото отклонение, което непозволява в неврона да не постъпва никакъв сигнал. Този тип архитектура се прилага при класификационни задачи.
- Многослойни рекурентни със *feedback* и/или *feed-forward* (прави връзки) механизъм на обучение

6.3.3. Обучение на ИНМ

Обучението на ИНМ е процес, при който мрежата сама извлича функционалните зависимости от подадените на входа дани, и като резултат помни теглата на връзките и самата структура. По този начин след определен брой цикли, време или до достигане на желано ниво на определен вид грешка системата се самообучава. при завършване на процеса мрежата трябва да покаже прогнозата, за която бъде запитане или класификацията, която се изисква да направи. Всеки цикъл на обучение, през който преминава обучащото множество се нарича епоха. Всяка *епоха* се различава от останалите по тегловните коефициенти и това предизвиква нов и различен отговор на средата, в която мрежата е поставена.

Видове обучение на ИНМ

- Според топологията на ИНМ:
 - обучение с постоянна типология (fixed topology)
 - обучение с адаптивна топология (dynamic topology)
 - модел на базата на принципа за конструктивизма
 - модел на базата на принципа за селективността
 - хибриден модел
- Според начина на добавяне на данни в ИНМ
 - Активно обучение - когато данните биват предварително селектирани, и само тази част, която се одобрява влиза в ИНМ за обучение
 - Пасивно обучение - всички данни участват в обучението на ИНМ
- Според продължителността на обучението
 - One time learning - обикновено обучението на ИНМ е еднократно, след което системата е готова да бъде използвана
 - Lifelong learning - някои системи са така проектирани, че на практика процесът на обучение е многократен и се извършва при всяка промяна на средата (информацията). Обикновено в такива системи периодично постъпват нови данни. Подходящ пример са системите за анализ на индекси, пазари, цени... Естествено такъв подход е приложим и в сферата на биологичните науки.
- Според начина на обучението:
 - Off-Line Learning [171]: това е най-разпространеният метод на обучение. Осъществява се чрез многократно разпространяване на множеството от данни през мрежата, като при всеки цикъл мрежата оптимизира структурата си (теглата), докато постигне целта или друго поставено условие за край на обучението. Представител на този вид обучение е алгоритъмът за обратно разпространение на грешката (Backpropagation) [172]
 - On-Line Learning[173][174]: метод на обучение, при който всеки пример се "научава" поотделно. Прилага се рекурсивно за всички примери в мрежата. На всяка стъпка от процеса структурата на модела и параметрите биват променяни в съответствие с "наученото". Тази адаптивност е характерна за биологичните системи.
 - Комбинирано обучение - симбиозата се осъществява чрез "хващане на малък прозорец" от данни от непрекъснат входен поток. Т.е. в рамките на прозореца от данни говорим за on-line обучение, т.к. прозорецът се движи по данните, обикновено със стъпка единица.
- Според това дали в процеса на обучение се използва критерий за коректност на резултата:
 - Надзоровано обучение, с учител (Supervised Learning)[175] - когато има критерий за коректност на процеса на обучение, т.е. прилага се оценка на

грешката, като се сравняват заложеният изход в примера и прогнозирания изход от системата. Т.е. поставянето на конкретна цел пред обучението за достигане на определено ниво на достоверност определя обучението като “надзиравано”.

- *Ненадзиравано обучение*, без учител (Unsupervised Learning)[175] - при този вид обучение няма как да се прецени до колко точен е резултатът. Характерно за методът е, че системата трябва да разполага с много голямо количество от данни, за да може сама да извлече характеристиките на данните, така че да образува кълстери от информация, на базата на които да проведе обучението. И докато при надзираваното обучение параметрите на системата плавно се променят, то тук се наблюдават големи скокове. Примери за такива системи са мрежите на Хопфийлд и асоциативната памет.
- *Подпомагано обучение* (Reinforcement Learning)[175] - тази стратегия е с оптимационен характер. Тя се позиционира между двете крайни стратегии - надзираваното и ненадзираваното обучение. Основното различие е използването на оптимационна функция, като се търси нейният максимум. Функцията отразява сумата от бъдещите стойности на ефекта от прилагането на конкретна стратегия (политика) от преминаването от едно състояние в друго. На практика ефектите се явяват теглата на отделните неврони, а стратегията определя насочеността на графа и наличието на рекурсия. За откриване на точната политика могат да се използват различни алгоритми: **Brute Force, Value Functions, Direct Policy Estimation** и др. Известни са задачите за въоръжения бандит [176], контрол над роботи, разписанието за асансьор, игра нашах и др. Всички тези задачи търсят баланса между експлоатацията на ресурсите и получаването на информация за бъдещите ресурси, т.е търси се решение на проблема exploration vs. exploitation
- *Състезателно обучение* (Competitive Learning)[177] - най-общо казано при състезателния принцип има само един който печели. Т.е. всеки входен неврон от ниво i се насочва към (определя) точно един изходен неврон от ниво $i+1$, който се обявява за победител. При тази постановка, със детерминирането на всяка следваща връзка, намаляват възможностите за избор на победител. Целите, които преследват при състезателното обучение са минизиране на грешката, максимиране на ентропията, картографиране на данните с цел визуализация. Такъв тип обучение е подходящ при решаването на класификационни задачи.

6.4. Генетичен Алгоритъм

Анализът на данни и създаването на прогнози често могат да бъдат формулирани като проблеми на търсенето – например: търсене на модел, който да обяснява данните, търсене на правила за правене на прогнози, или търсене за дадена структура или сценарий и прогнозата на данните. ГА се използва за решаване на такива проблеми като предсказване на структурата на протеини. Един от най-перспективните и бързоразвиващи се области на приложение е анализ на данни и прогнози в молекулярната биология. ГА са били използвани, при тълкуването на данни от ядрено магнитен резонанс, за да се определи структурата на ДНК [178], за намирането на правилната подредбата за неподредена група от ДНК фрагменти [179], [180] - асемблиране, и предсказване на протеинова структура.

Мелани Мичъл пише през 1995 г., че "В действителност, предвиждането на структурата на протеина от неговата амино киселинна последователност е един от най-важните нерешени проблеми на молекулярната биология и биофизика. " [181]. Това нейно твърдение, остава вярно дори и към днешна дата, въпреки че има редица публикации, опити и експерименти.

6.4.1. Биологическо описание на механизма и задачи, които решава

ГА решава оптимизационни задачи, като е свързана предимно с претърсването на дадено множество от данни, отговарящо на определени критерии.

Основен термин е "Хромозома"-та. Тя представлява набор от параметри, които подлежат на изследване. Всяка хромозома има своя "опашка", която съдържа категоризиращи и/или оценявачи признания/стойности за хромозомата. Казва се, че всеки параметър е бит от хромозомата. Следващата фигура е адаптация по [182].

Фигура 9: Обща схема на ГА



6.4.2. Генетични оператори

Т.к. ГА е алгоритъм инспириран от природата, и по-точно генетиката, то е логично и операторите, които използва да бъдат заимствани от апаратът на еволюционната генетична теория.

Mutation (Мутация) - Тя се състои в

случаен избор на бит в даден индивид x и смяна на състоянието му. В природата, вероятността за мутацията е малка и е предимно еволюционен или стрес фактор. Тъй като мутацията прави основни промени в хромозомите, получени от предходни генерации и може да се промени добро решение в лошо, тя трябва да се използва внимателно. Вероятността за мутация е параметър, който показва колко често ще се променят гените на хромозомите. Ако не е осъществена мутация, новото поколение се образува след операция кръстосване. Ако е извършена мутация, части от хромозомите са били променени.

Crossover (Кръстосване) - При този процес новият индивид се получава чрез деление на две изходни хромозоми на две или повече части и обмен между тях. В резултат се наследяват части от генетичния материал на двамата родители. Например кръстоската след 3-тия бит на

- родителските хромозоми:  и 
- дава като резултат:  и 

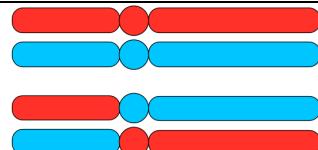
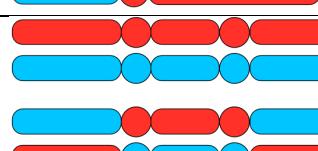
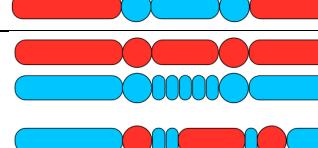
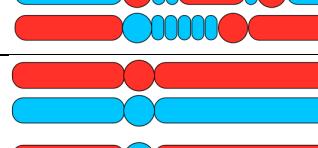
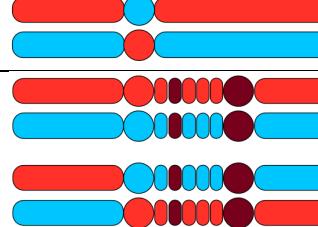
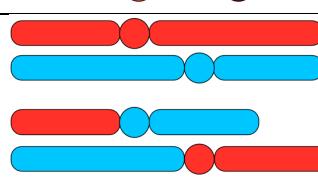
Кръстосване в една точка	
Кръстосване в две точки	
Еднотипно кръстосване Разменят се произволни "гени" след точката на кръстосване	
Кръстосване при кодиране по стойност Кръстосване при двоично кодиране Разменят се ст-те/битовете, в точката на кръстосване	
Кръстосване при пермутации Лилавите точки в двете родителски хромозоми имат еднакви ст-ти и не се копират при операцията.	
Кръстосване при дърворедно кодиране	

Таблица 2: Видове кръстосване

Inversion (Инверсия) - Инверсията има аналогично действие на конвертиране на един алел в друг. В биологията алелът представлява допустим и съществуващ вариант на даден ген. В случай, че ГА е бинарно кодирана, то алелна е само двойката (0;1).

6.4.3. Методи за селекция

Selection (Селекция) - това е начин за избор на родителските хромозоми, които да участват при формиране на новото поколение. За да се изберат най-добрите, всички хромозоми се подлагат на оценяване, доколко отговарят на изискванията на задачата (проблема), която се решава. След това се селектират най-добрите (годните). Съществуват множество подходи за селекция на хромозомите. Някои от тях са селекция по кръгова рулетка, състезателна селекция, селекция по ранг, селекция на устойчивите състояния и др.

✓ *Селекция по кръгова рулетка.* При селекцията по кръгова рулетка всяка от хромо-вомите участва със стойността на своята жизнено способност (годност). Колкото по-годна е една хромозома, толкова по-голям шанс има да бъде избрана. За онагледяване на метода ще си представим една диаграма тип „пай“, където са поставени всички хромозоми от популацията. Всека хромозома има собствено място, като големината ѝ („пая“ ѝ) съответства на нейната жизнеспособност. Изборът наподобява на хвърляне на топче в рулетката. Хромозомите, които са по-жизнеспособни, ще бъдат избирани повече пъти.

Основен недостатък на селекцията по кръгова рулетка е че, ако жизнеспособността на един от хромозомите е 90% от цялата кръгова рулетка, тогава останалите хромозоми ще имат малки шансове да бъдат избрани.

✓ *Селекция по ранг.* Селекцията по ранг преодолява недостатъка на селекцията по Кръгова рулетка. Селекцията по ранг първо подрежда хромозомите в популацията в съответствие с тяхната жизнеспособност. След това всяка от хромозомите получава ранг от 1 до N (N е броят на хромозомите). Най-жизнеспособната ще получи най-голям ранг (т.e N), а най-нежизнеспособната получава ранг 1.

✓ *Селекция на устойчивите състояния.* Идеята за селекцията на устойчивите състояния е голяма част от хромозомите да просъществуват в следващото поколение. Генетичният алгоритъм при този вид селекция работи по следния начин: Във всяко поколение се избират няколко хромозома с висока жизнеспособност, които ще участват в създаване на новото потомство. Избират се също така и няколко хромозоми с ниска жизнеспособност, които се премахват, и на тяхно място се поставя новото потомство. Останалата част от популацията оцелява и се записва отново в следващото поколение.

✓ *Селекция чрез прехвърляне.* При селекцията чрез прехвърляне първо се копират една или няколко хромозоми, които са с висока жизнеспособност, от старата в новата популация. Останалата част от новата популация се формира чрез кръстосване и мутация на копираните хромозоми. Този метод може драстично да увеличи времето за изпълнение на алгоритъма. тъй като чрез прехвърлянето се предпазват от загуба хромозомите с най-висока жизнеспособност.

✓ *Crowding* – това е селекция, при която новото поколение заменя най-сходното на него родителско. По този начин се избягва натрупването на прекалено много сходни поколения в селекцията [183].

6.4.4. Параметри на ГА

Параметрите на ГА дефинират характеристиките на използваните методи за селекция и оператори. Поддържат се: размер на популацията, процентни нива на операторите, които определят в каква степен всеки оператор може да влияе върху бъдещите поколения, тип на стратегията за селектиране.

Генетичният алгоритъм може да се прилага както самостоятелно, за решаване на определена задача, така като оптимизираща техника върху други методи на SC, например за определяне архитектурата на изкуствената невронна мрежа.

6.5. Подходи за отчитане на непълнотата на информацията

Биоинформатиката може да се разглежда като приложение на софт-компютинг методи и методи на статистиката върху данни с биологичен, биохимичен и медицински произход. Генерираната информация от технологиите за паралелно секвениране на геномни данни се отличава с големи обеми и интегрирана грешка от технологията за секвениране – продукт на самата апаратура. Анализът на такъв тип данни е свързан с характеристики като “непълнота” (до момента асемблираните данни генерират липси, спрямо регферентния геном, с който се сравняват), “неопределеност” (характерът на всеки един геном предполага наличието на формации като “повтори” и “хомополимери”, които се определят като противоречиви знания, многозначност, размитост), “неточност” (поражда се както от технологиите, с които се произвеждат данните, така и от факта, че всеки ген има своите разновидности. Поради това те са приблизително точни.). Всички тези характеристики, от своя страна описват областта на приложение на методите на софт компютинг.

Следващата таблица отразява по какъв начин се разглеждат понятията “Непълнота”, “Недостиг”, “Неопределеност” и “Неточност” на данните, каква е връзката между тях, от какво зависят и кои са източниците, които ги определят.

Таблица 3: Източници на непълнота и недостиг на данни

Тип на Непълнотата или Недостигът на данни	Зависи от:	Източници
Неопределеност	съдържанието на информацията (данните)	<ul style="list-style-type: none"> • Непълни или противоречиви знания • Размити граници на понятия • Твърдения с многозначна скала на истинност
Неточност	достоверността на информацията (данните)	<ul style="list-style-type: none"> • Когато данните не са прецизни • Когато данните са от ненадежден източник или се характеризират като вероятно грешни, т.е. като несигурни • Когато данните са с приблизителна точност

Подходите, които се използват при отчитане на непълнотата на информацията са представени на следващата схема:

Фигура 10: Подходи за отчитане на непълнотата на информацията



6.5.1. Вероятностен подход

Вероятностният подход се характеризира с това, че се прилагат детерминистични разбирания и логика при отчитането и анализа на несигурността на информацията (данныте). Тук вероятността има два аспекта, в които се разглежда:

а) като обективна величина - В основата на класическата вероятност стои понятието "случай". Случаите са събития, които са равновъзможни и цялото пространство се състои от краен броен брой случаи. Вероятността на всеки от тези случаи е една и съща. Счита се, че вероятността дадено събитие да бъде вярно ($P(A)$) е отношението между броя на благоприятните случаи m и общия брой на възможните случаи, т.е $P(A)=m/n$

б) като субективна величина - разглежда се като субективно очакване дадено събие да се случи. В този смисъл когато се говори за изкуствен интелект или за анализ на реални системи⁹, тази интерпретация е по-подходяща.

Вероятностните мрежи се базират на апарат от формули на условната вероятност.

Метод на Бейс и Вероятностни мрежи

Мрежата на Бейс[184] представлява графичен модел, който кодира вероятностни връзки между променливите. Когато се използва заедно със статистически техники, графичният модел има няколко предимства при моделирането на данните:

- ✓ Моделът кодира зависимостите между различните променливи, дори в случаите когато липсват стойности за някои от променливите.
- ✓ Може да се използва за изучаване на корелацията между променливите, с цел при въвеждане на част от тях да се предвидят останалите.
- ✓ Семантиката на модела е едновременно и причинно-следствена, и вероятностна. Поради това, той е идеален представител на съчетанието на предварителни знания (което често идва в причинна форма) и данни за анализ.
- ✓ Едновременното използване на статистически методи на Бейс и мрежи на Бейс, предполага ефективен подход за преодоляване на *overfitting*¹⁰ на данните.

⁹ Под реални системи следва да се разбира всяка система, която подлежи на описание и анализ, и която е предмет на реалността, която ни обкръжава. Например: биологичните системи, различни игри, които изискват вземане на решение и други системи, описвани като случайни.

¹⁰ Overfitting - модел на данните, при който и най-малките промени в изходната информация оказват голямо влияние, поради което прогнозите, които се произвеждат не са верни. Overfitting мрежите познават отлично своите данни, защото при тренинга дори и най-малкия шум е изчистен. Съществуват специални Robust алгоритми, които целят да предпазят мрежата от трениране на зашумените данни. Признак, че дадена мрежа да се окаже overfit-ната е ситуацията, при която грешката от тренирането е монотонно намаляваща, а валидиращата грешка нараства.

Вероятностните мрежи са едно от най-популярните приложения на метода на Бейс, известно още като мрежи на увереност. Тези мрежи показват с каква вероятност всяко събитие от мрежата ще настъпи в зависимост от предходните събития, като всяко събитие представлява променлива (variable) с минимум две значения и минимум две стойности. Във вероятностна мрежа се представят зависимостите между отделните величини, като се задава съвместното вероятностно разпределение, което представлява n-мерна таблица, чиято сума е 1-ца, т.к. включва всички условни вероятности. Представя се като насочен ацикличен граф със следните характеристики:

- Случайни величини, формиращи възлите в мрежата.
- Насочени рамена, които посочват директното влияние на една величина върху друга.
- За всеки възел е дефинирана таблица със условни вероятности, която задава вероятността на дадена стойност на променливата, представляваща зависимост от възможните комбинации на стойности в родителските възли.

Основни компоненти на вероятностните мрежите:

- Правят се изводи за недиректно заложени променливи в мрежата – Веднъж построена, мрежата на Бейс (състояща се от знания, данни или тяхна комбинация), трябва да бъде определена, като се зададат вероятностите за всяка променлива. Мрежата се моделира напълно и поради тази причина може да бъде използвана, за да даде отговор, който ще да бъде частично верен/грешен, т.к. представлява вероятност за съдване. Този отговор не е инициализиран в системата, той е изчислим, имплицитно заложен. Този процес е известен като *probabilistic inference* (в пр. “вероятностен извод”). Методите за постигането на вероятностни изводи са: елиминация на променливи [185]; кеширане на поддърво от променливи [186]–[188], рекурсивно запитване [189].
- Изучаване на вероятностния модел на разпределение (*Probabilities learning*).
 - Модели с напълно дефинирани променливи – Свободно полиномно разпределение, което е обобщен случай на общото Биномно разпределение.
 - Модели с частично дефинирани променливи:
 - “Частичност, зависеща от състоянието”: Dirichlet Distribution

- “Частичност, незасеща от състоянието”: апроксимиращи методи от тип Монте-Карло, Гаусова апроксимация, Многовариантно Гаусово Разпределение, MAP¹¹ и ML¹² апроксиматори, Expectation-Maximization [190] алгоритъм.
- Изучаване разпределението на параметрите – подобно на изучаването на променлите и тук се прилагат цитираните по-горе апроксиматори и разпределения, като съществува вариант параметрите да бъдат третирани като променливи, но това усложнява много задачата.
- Изучаване на структурата – в идеалния случай, мрежата е конструирана с участието на експерти. Възможно е да се проведе автоматично обучение, като се използват насочени ациклични графи[191].

Метод на Демпстър-Шафър

Представлява обобщен модел на метода на Бейс, като се добавя “правдоподобни истини” [192] към мрежата на Бейс. Теорията е била развита Артър Демпстер [193] и Глен Шафер[192][194]. Т.е. може да се каже, че вероятността по метода на Бейс е \leq правдоподобната истина.

6.5.2. Други подходи при отчитане непълнотата на информацията

В литературата са известни са известни още и следните подходи за отчитане на непълнотата в данните:

- Коефициенти на достоверност
- Коефициенти на размитост
- Немонотонни логики
- Логика по подразбиране
- Модални логики
- Автоепистемични логики

6.6. Синергизми

Характеристиките на софт компютърните приложения, разкриват, че всеки инструмент има своите плюсове и минуси. Размитата логика дава приблизителни решения, ИНМ отговаря за машинно обучение, а ГА може да се използва за оптимизационни задачи, докато Доверителните мрежи могат да се използват за анализ на зашумени данни от различни сензори. Синергизмите представляват съчетаване на два или повече метода и съответно на техните

¹¹ *maximum a posteriori* (MAP) configuration from Gaussian distribution

¹² Maximum Likelihood

силни страни. По отношение на методите на размитите множества са възможни следните синергизми:

6.6.1. Размита логика и ИНМ

Синергизъм от този тип позволява:

- обучение въз основа на приблизителни знания
- изводи чрез използване на знанията, придобити в резултат на обучение чрез невронна мрежа.

6.6.2. ИНМ и ГА

Невронни мрежи и ГА заедно могат да доведат до различни форми на взаимодействие. В повечето случаи ГА се използва за оптимизиране на тегловите коефициенти на невронната мрежа или за определяне на архитектурата на ИНМ.

6.6.3. Размита логика и ГА

В общия случай ГА оптимизира параметрите, на размита система. А размита система използва fuzzifier за да нормализира сигнала в диапазон от [0, 1]. Кривата е наречена “разпределение за членство”. Изборът на разпределение има значително въздействие върху отговора на размита система. ГА има за задача да оптимизира принадлежността към дадена група (клъстър).

6.6.4. ИНМ-Вероятностни мрежи

Това е сравнително нов тип на синергизъм. По дедукция следват два възможни вида синергизъм в тази категория:

- Невронните мрежи могат да бъдат използвани за определяне на условните вероятности във вероятностната мрежа.
- Сливане на мулти-сетивни данни чрез изчисления по метода на Демпстър - Шафер, като резултатът може да бъде използван при надзорувано обучение на невронната мрежа.

6.6.5. ГА-Вероятностни мрежи

Отново по дедукция, тук следва да се предположи, че ГА може да се използва, за да се адаптират условните вероятности на мрежата, като критерий за достоверност е възможно да се използва разликата действителните и априксимираните вероятности.

6.6.6. ИНМ-Размита логика-ГА

Един от възможните варианти е:

- ИНМ като модел класификатор, обучен с размити дистрибуции за членство, предварително настроени от ГА
- ГА може да се използва в тясно «силна» невро - размита система за оптимизиране на параметрите ѝ.
- ГА може да се използва за да се определи най-добрият набор от обучителни примери при «силна» невро - размита система.

1. Постановка на задачата

Всяка задача поставена пред информатиката е свързана с използването на математически методи и модели за нейното решение. Когато говорим обаче и за анализ на данни, към този математически комплекс се прибавят статистически подходи, които помагат разбирането на характера на данните. Задачи се поставят от различни области науката като физика, химия, биология, генетика, геномика, протеомика и т.н. По правило, научната област, която поставя задачата дава и ограниченията, с които решението трябва да бъде съобразено.

В нашия случай, данните представляват кодиращи ДНК последователни, секвенирани с NGS технология на платформата 454 на Roche. Последното изречение въсъщност определя някои от най-важните условия, скоито следва да се съобразяваме: 1) дължината на секвенирана последователност е с размер около 300-400 базови нуклеотида, 2) изходните данни за обработка и анализ съдържат фоново зашумяване, дължащо се както на самия процес на секвениране, така и на особеностите на платформата. Основната задача е определена като асемблиране на секвенирания геном, т.е. трябва да се съставят дълги последователности от кодиращите ДНК къси прочити, които в най-добрия случай да представляват определен ген. Става ясно, че кръгът от поставените задачи се разширява, както се появяват и нови ограничения:

- Подзадачи – филтриране на зашумяването; сравняване на късите прочити; съставяне на консенсусни секвенции
- Допълнителни ограничения – поставяне на критерии за достоверност, както при изчистването на шума, така и след асемблирането на късите прочити

Методите от информатична гледна точка, които се използват за решаване на задачите са от областта на софт-компютинга – невронни мрежи, генетичен алгоритъм, графи, кълстеризиационни методи, дистанционни матрици, статистически анализ за достоверност, графични методи за представяне на анализираната информация – роза на ветровете, стандартни линейни и стълбови графики.

Както беше отбелязано в началото, биоинформатиката е мултидисциплинарна наука, която служи за анализиране на последователностите, откриване, анотиране и картиране на генни последователности, изясняване на функциите

им, построяване на филогенетични дървета и проследяване на родствени връзки и дистанции между видовете, определяне на принадлежността на секвенирани организми към съответен клас. В контекста на т.нар. паралелно поколение секвениране, както и на по-новите технологии за секвениране, задачите на биоинформатиката вече се разпростират и върху асемблирането на получените данни под формата на къси прочити, като основната цел е постигане на консенсусни секвенции, които да могат да бъдат успешно анотирани. Характерно за секвенаторите, различни от първо поколение е, че те първо генерираят известно количество грешки, и второ предоставят секвенирания материал под формата на множество секвенции. Това обуславя необходимостта от прилагането на биоинформатични методи и в областта, касаеща изчистването на шума (грешките) от данните.

Основните задачи, които стоят пред почти всеки биоинформатичен анализ включват три групи софтуерни инструменти: бази данни и методи за търсене на секвенционна информация, съпоставяне на секвенции, асемблиране на данни от паралелни технологии за секвениране.

2. ДАННИ

2.1. Данни от паралелно геномно секвениране

През последните почти три десетилетия разработването на методите за секвениране на ДНК промени коренно представите за организацията на геномите на различните видове организми. Те позволиха определянето на нуклеотидната последователност на голям брой гени и дешифриране на функцията, която те изпълняват в генома. Информацията за полиморфизмите на ниво ДНК позволи използването им като маркери за генетично картиране, улесни значително клонирането на гените, спомогна за по-дълбоко навлизане в същността на еволюционните взаимовръзки и постави началото на редица проекти по изследване на генетичното разнообразие. В настоящата глава се разглежда информацията, получавана от секвениране с кратки описания на технологичните процеси и областите на приложения, а в дискусията се обсъждат техните предимства и недостатъци и задачите, които поставят пред биоинформатиката, като съвременна наука, която подсигурява методите на математически и статистически анализ на данните, имплементирани под формата на софтуер за анализ.

През годините са създадени различни технологии на секвениране, като за първото поколение е възприето да се казва, че това е традиционният начин, при който получаваме секвенци на отделни гени от съответния геном. Този тип секвениране се оказва, че е доказано най-точният и в същото време най-бавният и скъп процес в сравнение със последвалите поколения секвенирния (2-ро, 3-то и 4-то), които за краткост ще наричаме “Следващо поколение секвениране” и “Паралелно геномно секвениране”, въпреки че в литературата възприетият термин “Next Generation Sequencing” се отнася предимно за втората генерация машини, които предлагат паралелно секвениране. Характерното за всички “следващи”, е че произвеждат множество от секвенции с определена дължина, която е предмет на конкретната технология.

Тук по-подробно ще бъдат представени са технологиите за секвениране на NGS данни с платформите Sanger и 454, т.к. данните, които са използвани за анализ и асемблиирани са продукт именно на тези технологични концепции. Всички останали технологии са маркирани като достижения на научно-техническия прогрес, но са представени на кратко.

2.2.Биологично представяне на данните

Технологии за получаване на секвенционни данни

СЕКВЕНИРАНЕ I-ВО ПОКОЛЕНИЕ: Секвениране с терминиращи нуклеотиди
Наименование: Дидеоксинуклеотиден метод на Sanger за секвениране на ДНК. Най-често използваният в молекуларната биология метод за секвениране на ДНК е чрез прекъсване нарастването на новосинтезираната ДНК верига. Методът е създаден от F. Sanger през 1975г. При него се използват четири радиоактивно или флуоресцентно белязани дидеоксинуклеотиди (ddNTPs) - ddATP, ddGTP, ddCTP и ddTTP, които нямат хидроксилна група на 2'- и 3'- място, необходима за формирането на фосфодиестерна връзка между два нуклеотида. Поради това всеки от тях прекратява синтеза на новата полинуклеотидна верига върху ДНК матрицата.

Технологичен процес

Процедурата включва провеждане на ДНК полимеразна реакция в 4 различни епруветки. Всяка епруветка съдържа разтвор с четирите нормални деоксинуклеотиди (dATP, dGTP, dCTP и dTTP), ДНК полимераза, праймер и едноверижна ДНК матрица. Към всяка от епруветките се добавя само един от четирите белязани ddNTPs. Под действие на ДНК полимераза се синтезира верига, комплементарна и антипаралелна на ДНК матрицата. Дидеоксинуклеотидът във всяка епруветка се конкурира със съответния

нормален dNTP и се включва в нарастващата верига случайно, вследствие на което се прекратява синтезата на новата ДНК верига. Това прекратяване е случайно, но базата, при която свършва синтезата, е комплементарна на добавения специфичен ddNTP. В резултат се синтезират фрагменти с различна дължина. Новосинтезираните белязани ДНК фрагменти се денатурират и се разделят чрез гел-електрофореза (с разрешителна способност до 1 нуклеотид) в денатуриращ полиакриламид-уреен гел, така че секвенцията на ДНК може да се определи експериментално - по позициите на ивиците в гела. За целта, всяка реакция, белязана като A, T, G, C се нанася в 4 индивидуални позиции на гела. ДНК секвенцията може да бъде прочетена директно от рентгенов фильм. Секвенцията на интересуващата ни матрица е комплементарна на експериментално установената нарастваща верига. Разчитането на секвенцията от рентгенов фильм става на основата на относителната позиция на фрагментите, различаващи се с дължина от един нуклеотид в четирите позиции, означени като A, T, G, C на гела в посока от долу нагоре.

Област на приложение

Този метод, както и автоматизираните високотехнологични секвенатори са в основата на много от проектите по секвениране на геномите на различни растителни, животински видове, в това число и човека.

СЕКВЕНИРАНЕ II-РО ПОКОЛЕНИЕ

Платформи за геномно секвениране, основаващи се на амплификация

Платформите за NGS, базирани на амплификацията, продуцират къси (35-100bd при Illumina и ABI SOLiD и 250-500 базови двойки (бд) при Roche 454 GS FLX) ДНК фрагменти, обозначени като „секвенционни прочити“ или „тагове“¹³. Независимо от някои технически различия тези платформи показват сходство в приготвянето на ДНК библиотеките за секвениране, изразяващо се в:

- 1) Подготовката на ДНК матрицата за секвениране
- 2) Лигиране на адаптори, съдържащи специфични последователности към двета края на предварително фрагментираната ДНК матрица, чийто избор зависи от използваната платформа за секвениране.
- 3) Намножаване на фрагментите на основата на традиционния PCR, чрез използване на адапторната секвенция като начална точка за стартиране на

¹³ В авторските разработки тези секвенционни прочити ще се разглеждат като символни низове.

амплификацията или като матрица за хибридизация. В зависимост от платформата се прилагат емулсионен и мънистен PCR.

Известните до момента NGS технологии използват отчитането на светлинен сигнал, отделящ се при добавяне на точния, комплементарен на съответния му в подложената на секвениране ДНК матрица нуклеотид (или олигонуклеотиди при SOLiD) в секвенционната реакция. По този начин изходните първични данни представляват записи от образи на светлинни сигнали, излъчени от всяка една от единичните, паралелно протичащи реакции на секвениране по време на всеки цикъл на секвениране.

Пиросеквениране - Roche (454) GS FLX

Платформата е създадена през 2004 г. От 454 Life Sciences Co., по-късно придобита от Roche [195]. През 2005 г. излиза първия доклад за използването ѝ от [196]. Тя работи на принципа на пиросеквенирането, в чиято основа е определянето на количеството на пирофосфата, отделен след присъединяване на един нуклеотид към 3' края на растящата ДНК верига. Тъй като при този метод се отчита сигнала, излъчван от количеството "отпаден продукт", а не от новосинтезираната ДНК (както е при останалите методи за секвениране, основаващи се на технологията на Sanger), то той не се нуждае от радиоактивни или флуоресцентно белязани нуклеотиди. През 2012 г. е извършено надграждане на платформата, и тя вече се казва Roche (454) GS FLX+. Подобренията засягат като броя, така дълбината на секвенираните прочити, съответно 1 милион и 1000 bp [197]. Когато говорим растения, то трябва да отчетем, че Roche 454 е вече добре настроена и оптимизирана по отношение на секвенирането на транскрипти, като увеличаването на дълбината на късите прочити допринася значително за подобряването на процеса по асемблиране [198]–[200], и води до съществено подобряване на идентификацията на нови транскрипти в добре познати и характеризирани геноми като този на A. Thaliana [201]. Платформата на Roche – 454, е добре настроена дори за непознати геноми (*de novo*) на растения [202]. Геномите на краставицата (*Cucumis sativus*, 376 Mb) и този на A. Thaliana (157 Mb) бяха секвенирани като доказателство за качество на платформата 454 FLX Titanium, която използва "сдвоени" библиотеки, които лежат в основата на процеса по асемблиране [203].

Технологичен процес:

Полимеразната реакция се провежда стъпалообразно в един реакционен съд (ямка), в който на всяка стъпка се добавя само един от четирите нуклеотида (A, G, C, T). Когато към растящата верига се присъедини един нуклеотид, то в

този случай се отделя само 1 молекула пирофосфат, но ако се присъединят последователно няколко еднотипни нуклеотиди (например ААААА, в случай, че матрицата съдържа ТТТТТ), то количеството на отделения пирофосфат ще бъде пропорционално на броя на еднотипните нуклеотиди. Пирофосфатът се определя количествено чрез фотохимична реакция, катализирана от ензимите - луцифераза и ATP-активираща суфорилаза. Photoхимична реакция, при която се отделя квант светлина, протича при наличие на луциферин, ATP и пирофосфат. За количественото ѝ определяне се използва фиброоптична система с възможност за мониторинг на множество паралелно протичащи реакции. Геномната ДНК се фрагментира неспецифично с рестрикционни ензими или се накъсва механично и се фракционира в агарозни гелове, при което се извличат фрагменти с дължина не по-голяма от 300-800 бд. След стъпка на „полиране”, включваща запълване на стърчащите краища на извлечените фрагменти със свободни нуклеотиди на принципа на комплементарност, към последните се лигираат два адаптора (A и B), представляващи къси синтетични олигонуклеотиди. Тези адаптори служат като начална точка за стартиране, както на амплификацията, така и за секвениране на селектирания библиотека от фрагменти. Единият адаптор (B) съдържа 5' биотинов таг (еталон или къса секвенция белязана с биотин) за имобилизиране на ДНК библиотеката върху покрити със стрептавидин монодисперсни зърна. След репариране на скъсванията, небиотинилираните едноверижни ДНК-и се освобождават и в последствие служат като ДНК матрици за секвениране. Чрез нано-титруване се определя качеството и оптималното количество (ДНК копия) на така получената библиотека от едноверижни ДНК фрагменти за провеждане на емулсионен PCR (emPCR).

Библиотеката от едноверижни ДНК фрагменти се имобилизира върху монодисперсни мъниста, така че всяко едно от тях да съдържа прикрепена към него единична, едноверижна ДНК молекула. Така заредените мъниста, заедно с амплификационните реагенти (ДНК полимераза, свободни нуклеотиди, PCR буфер) се емулгираат във водно-маслена емулсия за провеждане на PCR. Всяко мънисто представлява микрореактор, в който в резултат на PCR амплификация се получават до 1 000 000 копия от една ДНК молекула (фрагмент). След денатурация, получените едноверижни ДНК фрагменти остават свързани към повърхността на мънистото. В края на реакцията емулсията се разрушава и мънистата се натоварват чрез центрофугиране върху пикотитърплейт (PicoTiterPlate). Ямките на тази пикоплейт имат формата на восьчни клетки, като в една килийка може да попадне само 1 мънисто, натоварено с едноверижна ДНК. Към всяка от килийките, съдържаща мънисто се добавя ДНК инкубационна смес (с ДНК полимераза) и ензимни микрограмули

натоварени с ензимите луцифераза и АТР – активираща се сулфорилаза. Така подготвената микрореакторна система се центрофугира, за да се позиционират точно мънистата, натоварени с единична едноверижна ДНК по стените на килийките при секвенирането. След тази операция пикотитърплейта се поставя в специална реакционна камера на Genome Sequencer FLX Instrument. Позиционирането на микрореакторната система е такова, че дъното ѝ, което е прозрачно да опира до фотодатчик със същата плътност на фоточувствителните елементи, каквато е и тази на килийките. Отворите на килийките са обърнати към камера с циркулиращи разтвори (буфери за секвениране и нуклеотиди). Секвенирането във всяка килийка започва със хибридизация на ДНК праймерите (донори на 3'ОН група за ДНК полимераза), след което през реакторната система се пропускат последователно разтвори, съдържащи ДНК полимераза, луциферин, АТР и само един от 4-те деоксинуклеотиди (dATP, dTTP, dGTP, или dCTP). Времето за дифузия на реагентите от повърхността до дъното на килийките е 10 секунди, а това за протичане на реакцията е по-малко от 1 секунда, с което общото време не надвишава 10 секунди. Реакцията се повтаря със следващ нуклеотид. Фибооптичният датчик (CCD камера) измерва излъчената светлина от всяка отделна клетка. Като се има предвид, че броят на килийките в една реакционна плоча е 1 500 000, това означава, че едновременно могат да протичат 1.5 млн. отделни секвенционни реакции. На практика броят на реакциите е по-малък тъй като не повече от 30-40% от килийките са заредени с мъниста с прикрепена към тях едноверижна ДНК. Силата на сигнала, детектиран от фотодатчика е пропорционална на броят на нуклеотидите. Данните се съхраняват в стандартен флуограмен формат (SFF files) за последващ анализ.

Област на приложение

Технологията се прилага както при ресеквениране, така и при *de novo* секвениране на геноми и транскриптоми, а съществуващият софтуер организира процеса на асемблиране на бактериални геноми, геноми на инсекти, растителни геноми; технологията е приложима и при транскриптоми със средна дължина (< 3 Мб) и 16S рРНК в метагеномиката.

Секвениране чрез синтеза - Illumina Genome Analyzer

Концепцията за „секвениране чрез синтеза“ (SBS) предложена като патентна реализация от фирмата Solexa (UK), която впоследствие намира масовото си приложение чрез платформата за секвениране на фирмата Illumina/Solexa - Illumina Genome Analyzer е въведена през 2006г. Тази технология за

секвениране продуцира къси секвенции с дължина от порядъка на ~32–40bd (75-150bd при последните модели секвенатори на Illumina) от десетки милиони едновременно амплифицирани ДНК фрагменти.

Област на приложение

Технологията се използва при откриване на полиморфизми /варианти/ при ресеквениране на цели геноми; цели транскриптоми и откриване на гени в метагеномни изследвания

Процес на секвениране, катализиран от ДНК лигаза - Applied Biosystems SOLiD

Практическото приложение на секвенатора на фирма Applied Biosystems известен като Applied Biosystems SOLiD датира от октомври 2007г. За разлика от описаните по-горе секвенатори от тип следваща генерация той използва уникален процес на секвениране, катализиран от ДНК лигаза. Секвенирането със SOLiD (Sequencing by Oligo Ligation and Detection – Секвениране чрез Олиго Лигиране и Детекция) изисква около 5 дни и продуцира огромно количество бази данни за секвенции (фрагменти) от порядъка на 3-4 Gb (гигабайта/фрагмент) със средна дължина на секвенция 35 bd (при по-новите модели SOLiD 5500 и до 75 bd). Платформата на Illumina е широко използвана при секвенирането на растителни геноми. Примери за използването при *de novo* секвениране са работите на [204]–[209], а за ресеквениране на [210]–[212].

Област на приложение

Технологията се използва при откриване на полиморфизми /варианти/ при ресеквениране на цели геноми; цели транскриптоми и откриване (идентифициране) на гени в метагеномни изследвания

Процес на секвениране, използващ свързване с хистони - ChipSeq

Технологията е свързана с изследване, както на взаимодействията между протеините, така и на местата на свързване на ДНК с тях в генома. Методът позволява да се идентифицират функционалните елементи в генома: места на свързване на транскрипционните фактори, модификации на хистоните, хроматиновата структура и полимеразите в таргетния геном. Например, могат да бъдат определени местата на хистоновите модификации (H3K27me3, H3K4me2, H3K9me3 и др.), на свързване на Polycomb–група протеини

(PRC2:Suz12 PRC1:YY1) и Triorax-група протеини (Ash1) за изследване на епигенетичния профил или местата на свързване на РНК полимераза II за изследване профила на транскрипция.

Област на приложение

ChIPSeq е мощен метод за селективно обогатяване за ДНК последователности, свързани с конкретен протеин в живи клетки. Въпреки това, широкото използване на този метод е ограничено от липсата на достатъчно силен метод за идентифициране на всички обогатени ДНК последователности. Процесът **ChIPSeq** обогатява специфични омрежени ДНК-протеинови комплекси, като се използва антитяло срещу интересуващия ни протеин.

Платформи за транскриптомно секвениране

Регулирането на генната експресия е в основата на сложните взаимовръзки между генотипа и фенотипа на организмите. Процесите синтез и зреене на РНК са взаимосвързани и се координират и регулират от сложна мрежа от експресиращи се гени, отговорна за протичането на биологичните процеси. Успешното съчетаване на консервативността и пластичността на тази сложна мрежа от транскрипти е в основата на адаптивността на организмите към условията на средата. Задълбоченото разбиране на принципите и механизмите на регулиране на сложната експресия на гените е от съществено значение за дешифриране на причините за раковите заболявания и на други генетично-обусловени болести при човека, а също и за дешифриране на механизмите, отговорни за адаптивността и толерантността на растенията и други организми към стресови фактори. Въвеждането на високо-технологичните платформи за секвениране на ДНК (NGS) след 2005г. направи революция в транскриптомните изследвания, позволяйки провеждане на количествен анализ на РНК. Този анализ се основава на паралелно (едновременно) секвениране на кДНК библиотеки, получени чрез обратна транскрипция на РНК молекулите, представени в клетъчната популация или тъканта, която се изследва и поради това понастоящем е известно като РНК секвениране(RNA-seq).

Подготовката на библиотеката от фрагменти (транскрипти) за РНК секвениране е ключов фактор, тъй като тя отразява до каква степен получените данни за кДНК секвенциите отговарят на оригиналната РНК популация в клетката, тъканта или организма на определен етап от развитието му.

Технологичен процес:

Понастоящем са публикувани 6 протокола за РНК секвениране, които

позволяват запазване „специфността” на всяка от ДНК веригите. Те се различават по начина на лигиране на адапторите за кДНК и включват:

1. директно лигиране на РНК адаптори към РНК пробата преди провеждане на обратната транскрипция;
2. добавяне на адаптори при „обръщане” на веригата по време на обратната транскрипция;
3. използване на два случаини праймера (double – random priming), съчетано с екстракция върху твърда повърхност.
4. директно лигиране на ДНК адаптори за едноверижна кДНК;
5. обратна транскрипция на *ин витро* полиаденилирани РНК фрагменти, последвано от лигиране;
6. инкорпориране на dUTP по време на синтезата на втората верига на кДНК и разрязване с ензима урацил-N-гликозилаза.

Всяка молекула „с” или „без амплификация” се секвенира с една от комерсиалните платформи за секвениране, при което се получават секвенционни прочити от единия или от двата края на тази молекула. Секвенираните фрагменти са с дължина от порядъка на 30-400 бд в зависимост от използваната ДНК технология. По принцип, за секвениране на РНК (RNA-Seq) може да се използва всяка една от по-горе NGS технологии. След секвениране, получените секвенции се сравняват с референтен геном или референтни транскрипти или се асемблират *de novo* без предварителна информация за геномните последователности, при което се създава транскрипционна карта, отразяваща структурата на транскриптите, и/или нивото на експресия на всеки един от гените.

Област на приложение:

Последните две години бележат прогрес и в секвенирането на некодиращите РНК-и - малките РНК-и с дължина около 17-24 нуклеотида (micro RNA - miRNA) и късите интерфериращи РНК-и (siRNA), които играят важна роля в регулирането на генната експресия чрез модулиране на транслацията и стабилността на мРНК. Секвенирането им позволява определяне нивата на експресия на miRNAs и siRNAs и тяхната функция на основата на експресионния профайлинг, идентифициране на различни секвенционни изоформи, предсказване на нови miRNAs и siRNAs и на потенциалните им таргетни места в мРНК-и.

СЕКВЕНИРАНЕ III-ТО ПОКОЛЕНИЕ: секвениране на единични молекули

HELICOS BIOSCIENCES

Първата платформа за секвениране на единични молекули HeliScope е разработена от фирма Helicos BioSciences¹⁴. Тя продуцира 1 гигабайт секвенции на ден и използва секвениране по време на синтеза. Секвенираните фрагменти са с дължина до около 45-50 бд.

Област на приложение:

Helicos платформата е демонстрирана при секвенирането на дългия 6407 бази геном на бактериофага M13, с което са показани потенциалът и техническите трудности на новите методи за секвениране чрез синтеза, базиращи се на единични ДНК молекули. Чрез двукратно секвениране на ДНК може да се повиши точността на резултатите, а с въвеждането на нов тип терминатори (наречени виртуални) фирмата твърди, че намалява риска от неколократно включване на един и същ нуклеотид поради прекомерна активност на полимеразата.

VisiGen

Тази платформа използва модифицирана с флуоресцентна донорна молекула ДНК полимераза.

Област на приложение:

Компанията предполага, че тази платформа ще използва общ паралелен арей от навързани ДНК полимерази, които генерират $1-10^6$ бд секвенции/секунда, което изисква и много по-мащабни ИТ решения за управляване, съхраняване и обработка на такива обемни масиви от секвенционна информация.

PacificBio

Платформата за секвениране използва dNTPs с флуоресцентно белязани фосфатни групи. Получените секвенции са с дължина дори над 4000 бази, когато като матрица за секвениране се използва едноверижна кръгова ДНК молекула. Наблюдаваните грешки, включително делеции, инсерции и неправилно включени бази могат да бъдат елиминирани чрез създаване на консенсусна секвенция, произхождаща от множеството обиколки (многократно секвениране) на ДНК матрицата.

Област на приложение:

Подходяща е при секвениране на цели транскриптоми. Добре се допълва с другите платформи, насочени към ресеквениране с оглед откриване на секвенционни варианти /полиморфизми/ и хаплотипове.

¹⁴ <http://www.helicosbio.com>

Платформи за секвениране на ампликони

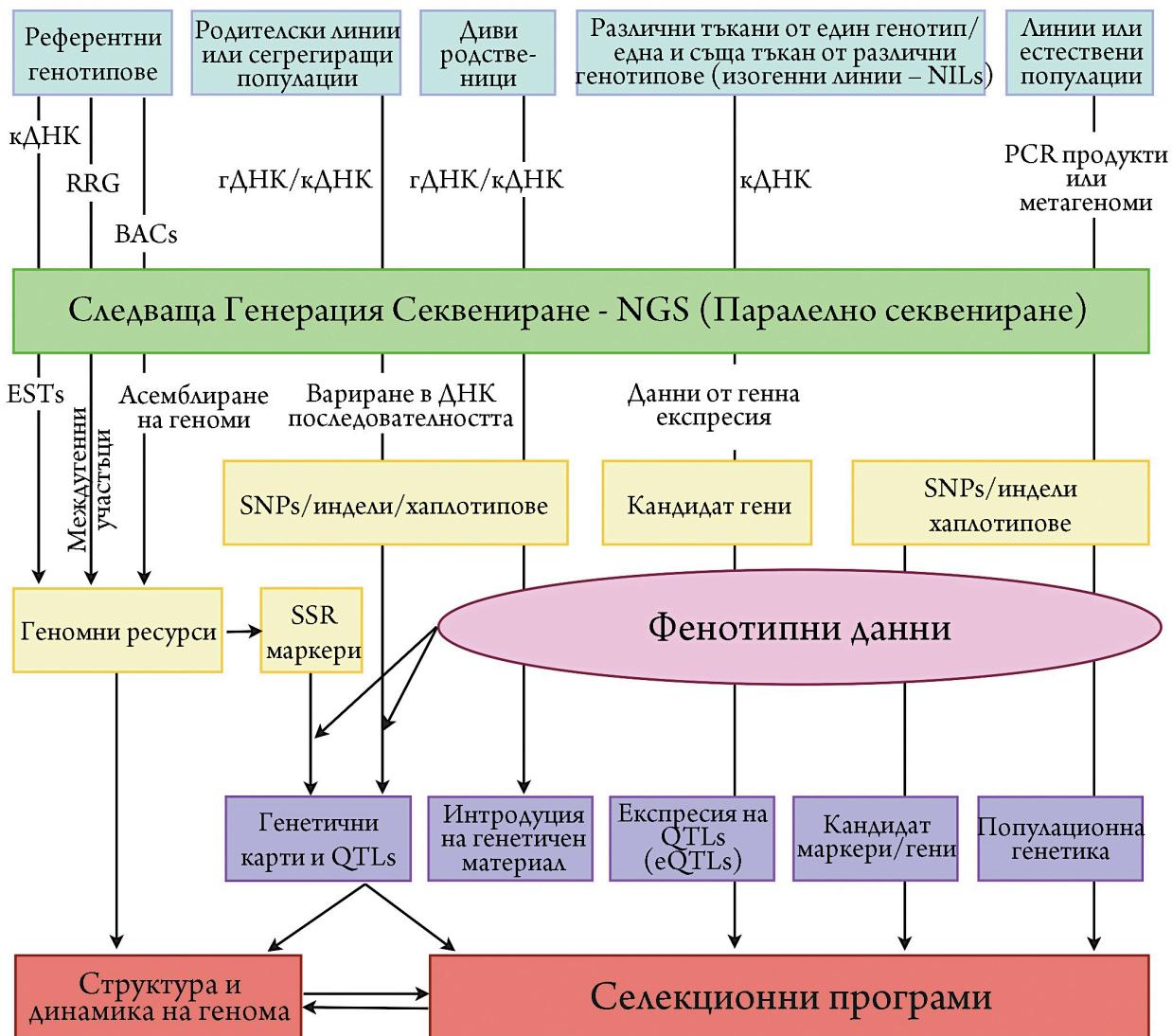
Секвенирането на ампликони (**Amplicon Sequencing**) е нова област на приложение на 454 технологията и позволява детекция на мутации с много ниска честота в специфични, амплифицирани на основата на PCR ДНК участъци. Този метод позволява идентифициране на соматични мутации в преби от пациенти, болни от рак или на редки нуклеотидни варианти при болни от СПИН. Геномният секвенатор 454 FLX (454 GS FLX) предлага софтуер (GAmplicon Variant Analyser), който автоматично налага получените от секвенирането секвенционни прочити срещу референтна такава (геномна или транскриптомна), позволявайки по този начин бърза и ефективна детекция на мутациите.

Данните от секвениране и задачите, които поставят за решаване

Технологичният сектор в областта на секвенирането се развива с бързи темпове. През последните години се появиха и развиха различни нови, по-бързи, по-ефективни платформи за секвениране, както на геномни и транскриптомни последователности, така и на секвениране на отделни молекули.

Платформите за паралелно секвениране на свързани (mate-paired) и линейни ДНК фрагменти намират широко приложение при *de novo* секвенирането (секвениране на все още несеквенирани геноми), в ресеквенирането (повторно секвениране) на цели или определени ДНК райони на вече известни, секвенирани геноми на различни организми, за метагеномни и транскриптомни анализи и установяване на SNP (Single Nucleotide Polymorphism) маркери за създаване на генетични карти и за геномна селекция при растенията и селскостопанските животни, за доказване на интродуциран генетичен материал при междуродствени кръстоски при растенията, за изследване на епигенетичните промени при различни видове организми, за картиране на структурни пренареждания (промени в броя копия на гените, балансираны транслокационни скъсвания и хромозомни инверсии), а също така и за асоциативно генотипиране при човека, животните и растенията.

Данните от паралелно секвениране, поставят редица задачи, показани на следващата фигура, от гледна точка на целите, които изследователската група си е поставила.



Фигура 11: Области на приложение на данните от паралелно секвениране

Изследователят вече е поставен в ситуация на избор. Той може да предпочтете конкретна платформа, защото тя би предоставила данни във формата, от който той има нужда за последваща обработка и биоинформатичен анализ.

Предимства и Недостатъци на различните платформи за секвениране

Основното преимущество на NGS технологиите се изразява в използването на по-високи технологични платформи в сравнение със секвенирането, основавашо се на Sanger метода. Последните модели апарати позволяват генериране на повече от 100 гигабайта секвенционни прочити/секвениране в сравнение с това на първите автоматични секвенатори с капацитет 70-100кб. Това се постига чрез едновременно паралелно секвениране на стотици хиляди или милиони фрагменти. Второто преимущество се изразява в елиминиране на клонирането и заместването му с PCR амплификация на ДНК фрагментите - стъпка, която напълно отпада при най-новите NGS технологии, основаващи се

на секвениране на единични ДНК молекули. Редуцирането на реакционния обем и едновременното секвениране на стотици хиляди фрагменти значително понижава себестойността на реагентите и като следствие на целия процес на секвениране. Поради непрекъснатото усъвършенстване на тези технологии през последните 10 години цената на секвенирането се редуцира от 10 000\$/Мбд секвенция до около 60\$ със 454 и около 1-2\$ при последните генерации апарати, които не изискват оптична система за детекция.

Платформа	Предимства	Недостатъци
Roche/454's GS FLX Titanium	По-дълги секвенции, Подобряване на възможността за разчитане на хомополимерни участъци, бързина	Скъпоструващи консумативи, висока честота на грешки в хомополимерни участъци
Illumina/ Solexa's GAII	Най-широко използваната понастоящем платформа	Ниско ниво на мултиплексиране на пробите – едновременен анализ
Life/APG's SOLiD 3	Двубазовият код позволява коригиране на грешките	Продължително време на секвениране
Helicos BioSciences HeliScope	Отсъствие на отклонение в представителността на матриците за геномни и секвенционни анализи	Високо ниво на грешки в сравнение с обратимите терминатори или химични реагенти използвани в другите платформи
Pacific Biosciences (target release: 2010)	Има най- висок потенциал за получаване на секвенции с дължина над 1кб	Най-високо ниво на грешки в сравнение с другите NGS химични реакции

Таблица 4: Сравнителна таблица за технологиите на секвениране

Независимо от преимуществата, посочените технологии притежават и някои недостатъци, които в отделни случаи редуцират приложението им. Тъй като повечето NGS продуцират къси секвенционни прочити, първоначалното им използване е било сведено главно до ресеквениране на геномите на някои щамове и сортове с цел идентифициране на мутации. Повторените райони, които се срещат с висока честота в растителните геноми са друг основен проблем, поради трудното им асемблиране, особено в случаите когато дължината им надвишава средната дължина на прочита.

Макар, че при NGS отсъстват стъпки на клониране, конструирането на библиотеките от ДНК фрагменти е трудно и включва няколко етапа – фрагментиране, лигирание на адаптори и PCR амплификация. Всяка една от тези стъпки води до индуциране на известно зашумяване в представителността на ДНК фрагментите и артефакти, поради което е повече от необходимо те да бъдат правилно отчетени. Например, дължината на фрагментите трябва да бъде

контролирана и оптимизирана за ефективно секвениране в зависимост от технологията.

Други ограничения са грешките и артефактите от секвенирането, като варирането в количеството им при различните технологии е различно. До момента, секвенирането, основаващо се на Sanger метода е с най-висока точност поради прецизността на *E. coli* полимеразата и се възприема за „златен стандарт“. Грешките, наблюдавани при секвенирането на къси хомополимери с 454 технологията например, се дължат на множество инкорпорирания на един и същ нуклеотид, вместо на единичен по време всеки цикъл от секвенирането. Модифицираните нуклеотиди могат също да са причина за неправилно инкорпориране или за блокиране на включването на следващия белязан такъв, ако флуоресцентния таг не е бил напълно отстранен. Грешки също се интродуцират при секвениране с ABI SOLiD, въпреки че тук всеки нуклеотид се отчита два пъти. Други проблеми са: отслабване на флуоресцентния сигнал и дефазиране (фазова разлика) на сигнала между паралелно секвенираните фрагменти. Това поражда високо ниво на грешки, особено в края на секвенцията. За тази цел компаниите са разработили методи (експериментални протоколи и софтуер), които непрекъснато биват усъвършенствани с оглед по-прецизния им контрол. От особено значение е точното отчитане и редуциране нивото на грешки при секвениране, насочено към откриване на редки варианти като SNPs и малки индели. В случая е повече от наложително да се осигури пълно покритие на генома и дори прилагане на подходяща статистическа обработка (използване на Bayesian метод). Това позволява грешките да бъдат отличени от редките варианти и да се елиминират. При наличие на референтен геном, късите секвенционни прочити могат да се съпоставят с него и показващите по-значително несъответствие да се отстраняват от по-нататъшни анализи. Алтернативно, могат да се комбинират данните от използването на няколко технологии за секвениране. Например, грешките в хомополимерите, генериирани с 454 могат да бъдат коригирани с използване на тези от Illumina и SOLid с по-ниско покритие, а в някои случаи и с тези от Sanger секвенирането (ако има такива). Следващата важна стъпка е да се асемблират късите прочити в по-дълги секвенции. Първоначално късите прочити се сравняват с референтните такива и се селектират само тези, които показват точно съвпадение. В случая това води до отстраняване на повечето от повторените секвенции и ограничаване на приложението на технологията само в проекти по ресеквениране на геномите. Това обаче е голям проблем при *de novo* секвенирането на дългите и сложни геноми, като тези на растенията. Въпреки, че компаниите предлагат софтуери за съпоставяне на късите прочити и референтната секвенция, през последните години се разработват нови

алгоритми за подобряване на асемблирането, което е и проблем номер едно при *de novo* секвенирането. Тези програми са адаптирани за съответните платформи (Newbler е по-подходящ за асемблиране на прочитите от 454, а Velvet - за данните от Illumina).

Както бе посочено по-горе един от основните проблеми при секвенирането на геномите е асемблирането на секвенционните прочити в по-дълги контиги. Решението на този проблем при метода на Sanger е да се секвенират фрагментите от двата им края (в сенс и антисенс ориентация) при използване на всяка една от двете вериги като матрица. Тази технология е залегнала в конструирането на ДНК библиотеката от фрагменти за NGS, основаваща се на секвениране на фрагменти със свързани краища. Последните могат да бъдат секвенирани с наличните NGS технологии и получените прочити да бъдат асемблирани в по-дълги контиги и в по-голями структури – мегаконтиги (scaffolds).

Високата производителност позволява използване на съвременните секвенатори не само за секвениране на една библиотека, но и на няколко или дори на 96 такива едновременно. В този случай е необходимо всяка от тях да бъде предварително маркирана (bar-coded) за последващо разграничаване на получената информация от секвенционни данни.

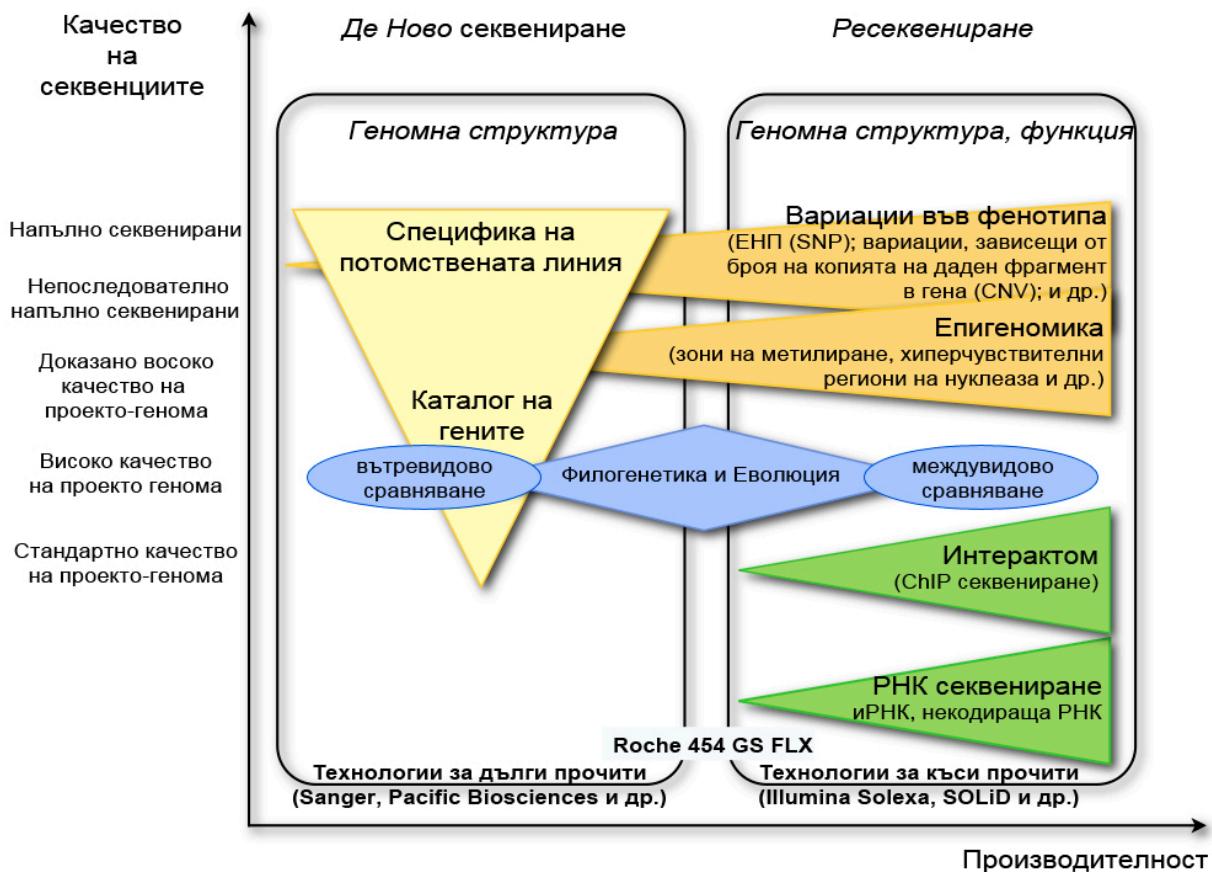
Макар и NGS да са изключително подходящи за ресеквениране на растителните геноми, през последните години тези технологии намират все по-успешно приложение и в *de novo* секвенирането. Особено улеснение при обработката на огромните масиви от секвенционни данни при *de novo* секвенирането е получените секвенции да бъдат сравнени с:

1) геномни или транскриптомни данни от секвенции на моделни или основни растителни видове, които са близкородствени с вида, чийто геном се секвенира и асемблира;

2) целия транскриптом или малки налични геномни бази данни от секвенции за съответния вид, получени на основата на различни платформи.

Този подход е бил използван и за откриване на маркери при човешкия геном и някои важни селскостопански култури.

Следващата фигура отразява изборът, който изследователят трябва да направи въз основа на областта на приложение, която представлява интерес за него.



Фигура 12: Платформи за секвениране, приложение, качество

Секвенционни бази данни и софтуер за анализ и асемблиране на NGS данни

В таблици № 4 и 5 са показани някои от основните бази данни за секвенции и по-често използваните софтуерни решения при сравняване на секвенциите, т.к. непрекъснато се генерираят нови решения.

Таблица 5: Секвенционни бази данни

Име	Описание
GenBank към NCBI (National Center for Biotechnology Information)	Една от трите първични бази данни, част от International Nucleotide Sequence Database Collaboration. Съдържа анотирани колекции от всички публично достъпни ДНК секвенции
ENA (European Nucleotide Archive) към EMBL (European Molecular Biology Laboratory)	Съдържа ДНК секвенции - асемблирани, анотирани или необработени. EMBL също е част от International Nucleotide Sequence Database Collaboration.
DDBJ (DNA Databank of Japan)	Третата първична база данни в International Nucleotide Sequence Database Collaboration. Съдържа ДНК секвенции.
UniProt (Universal Protein Resource)	Най-голямата база данни за белтъчни секвенции. Обединява белтъчните бази данни TrEMBL (към EMBL), SWISS-PROT (към SIB - Swiss Institute of Bioinformatics) и PIR (Protein

UniGene към NCBI	Съдържа данни от GenBank, клъстериирани по видове.
Ensembl	Геномна база данни за гръбначни и други еукариоти.
fRNAdB (Functional RNA DataBase)	Съдържа секвенции на некодиращи РНКи (miRNA; snoRNA и др.)
PlantGDB (Plant Genome DataBase)	Съдържа геноми на растителни видове

Съпоставянето на секвенциите е базата върху, която стъпват повечето от алгоритмите за геномна анотация, установяването на единични полиморфизми, определянето на процента на грешката от секвенатора, откриването на мотиви и т.н. Алгоритмите Smith-Waterman и Needleman–Wunsch са залегнали в почти всички варианти за съпоставяне. Съответно, най-широко приложение имат програмите FASTA и BLAST, като първата предлага локално, а втората – глобално сравняване. Използва се динамично програмиране, но само за тези секвенции в базата данни, които имат участъци, достатъчно сходни на изследваната секвенция. Методите, използвани за откриване на секвенции, които да отговарят на това условие, са изцяло евристични.

Следващата таблица представя най-широко използваните софтуерни решения в областта на сравняването на ДНК и РНК последователности.

Таблица 6: Софтуер за сравняване на секвенции

FASTA вариант	Описание	BLAST еквивалент
FASTA	Търси белтъци в белтъчни бази от данни и ДНК в геномни бази от данни. За белтъци по подразбиране $k=2$ ($k=1$ е с по-висока чувствителност). За ДНК по подразбиране $k=6$; 4 4 или 3 са с по-висока чувствителност; 1 е за много къси ДНКи.	BLASTP
SSEARCH	Използва Smith–Waterman алгоритъми претърсва белтъци срещу белтъци и ДНК срещу ДНК. По-чувствителен от FASTA при работа с белтъци.	BLASTN
FASTX FASTY	Претърсва белтъчни бази данни срещу ДНК след транслирането \square . FASTY е по-бавният, но по-добър вариант. BLASTX Установява дали дадена ДНК кодира белтък.	BLASTX
TFASTX TFASTA	Претърсва ДНК бази данни срещу белтък и се прилага основно за идентифициране на ESTs (expressed sequence tags). По-добър от FASTX, тъй като сравнението на белтъци е по-точно от TBLASTX ДНК.	TBLASTN TBLASTX
FASTF	Претърсва белтъчни бази данни срещу смес от пептидни секвенции (например получени чрез Едманова деградация).	

Основна задача, предхождаща същинския анализ на данни, получени от следващи поколения секвениране е процедурата по асемблиране на множеството от секвенции, продукт на секвенатора. Тук е представена една част от най-широко цитираните програми в тази област, като техният брой и подходите за реализация на такъв род задачи, непрекъснато расте.

Таблица 7: Софтуер за асемблиране и поддържани технологии и типове на обработваните данни

Софтуер	Тип	Технологии, които поддържа	Представен на	Тип на лиценза ¹⁵
ABySS	геноми	Solexa, SOLiD	2008	OK
ALLPATHS-LG	(големи) геноми	Solexa, SOLiD	2011	OK
<i>Celera</i> WGA				
Assembler / CABOG	(големи) геноми	Sanger, 454, Solexa	2004	OK
CLC Genomics Workbench	геноми	Sanger, 454, Solexa, SOLiD	2008	K
Cortex	геноми	Solexa, SOLiD	2011	OK
DNAexus	геноми	Illumina, SOLiD, Complete Genomics	2011	K
Edena	геноми	Solexa	2008	K
Euler	геноми	Sanger, 454 (Solexa ?)	2001	K/{HKЦ,A}
Euler-sr	геноми	454, Solexa	2008	HKЦ,A
Forge	(големи) геноми EST, метагеноми	454, Solexa, SOLID, Sanger	2010	OK
Geneious	геноми	Sanger, 454, Solexa	2009	K
Graph Constructor	(големи) геноми	Sanger, 454, Solexa, SOLiD	2011	K
IDBA	(големи) геноми	Sanger, 454, Solexa	2010	K/{HKЦ,A}
MIRA	геноми, ESTs	Sanger, 454, Solexa	1998	OK
NextGENe	(малки геноми)	454, Solexa, SOLiD	2008	K
Newbler	геноми, ESTs	454, Sanger	2009	K
Phrap	геноми	Sanger, 454	2002	K/{HKЦ,A}
TIGR Assembler	геноми	Sanger	1995	OK
Ray	геноми	Illumina, микс от Illumina и 454	2010	OK
Sequencher	геноми	Всички поколения секвениране	1991	K

¹⁵ OK – отворен код, K – комерсиална, K/{HKЦ,A} – комерсиална, но свободна за некомерсиални или академични цели, (...) – скобите показват най-вероятният лиценз, под който софтерът се разпространява

SeqMan NGen	(големи) геноми, exomes, транскриптоми, метагеноми, ESTs	Illumina, ABI SOLiD, Roche 454, Ion Torrent, Solexa, Sanger	2007	K
Velvet	(малки) геноми	Sanger, 454, Solexa, SOLiD	2007	OK

2.3. Формати данни от паралелно геномно секвениране

Съществува множество от файлови формати, разработени конкретно за данни от секвениране на ДНК. В настоящата работа са използвани следните файлови формати: fasta, sff, sra, fastaq.

Различни са мотивите, поради които в определени ситуации се предпочита един вид файлов формат пред друг. В следващата таблица са дадени описанията им, както и какви данни съдържат. Файловите формати са подредени според тяхната юерархия, в засимост от данните, които съдържат:

Таблица 8: Основни характеристики на файловите формати

Файлов формат	Архив	Готов за стрийминг при зареждане	Допълнителни данни	Мета данни	Компресия	Индексиране	Поименен парсинг на стринговете	Индексиране на имената на прочитите
SRA	ДА	ДА	ДА	ДА	ДА	ДА	ДА	ДА
SFF	НЕ	ДА	ДА	ДА	ДА	ДА	ДА	ДА
SRF	НЕ	ДА	ДА	ДА	ДА	ДА	ДА	ДА
Нативни формати ¹⁶	НЕ	НЕ	ДА	НЕ	НЕ	НЕ	ДА	ДА
FASTQ	НЕ	ДА	НЕ	НЕ	НЕ	НЕ	НЕ	НЕ
FASTA	НЕ	НЕ	НЕ	НЕ	НЕ	НЕ	ДА	НЕ

FASTA - Секвенционни прочити и идентификатори към всеки секвенционен прочит. Първият ред започва задължително със знака “>”. FASTA файловете, генериирани на базата 454 SRA файловете допускат 5 буквена азбука, където освен буквите A, C, T, G се появява и N-символизираща несигурност в значението (т.е. може да бъде някоя от другите 4 букви), освен това всички букви са главни.

Примерен изглед:

>EcoliK12 4300K bp
CCTTGTGCAGTAGCACTTAATCATCATGTTTAGCATTGATCTCTGCTCAATTCTT

¹⁶ Нативните формати се разработват специално за конкретна технология и производител на секвенционни данни. Поради тази причина те са с различен формат. Представят освен геномни/протеомни последователности и допълнителни данни, характерни единствено за конкретната платформа.

AAGCTAGACGCTCAATCTTCTTATGATGAACGATTCTTCATGGTGTTCATAT

.....

Предпочитан е, когато са необходими единствено секвенциите.

FASTAQ - Съдържа освен секвенциите и количествените характеристики за всяка една база. Първият ред за прочит задължително започва със знака “@”, следван от идентификатора на секвенцията. Вторият ред е самата секвенция, като за разлика от FASTA формата няма прекъсване на реда, т.е. една секвенция е представена на един ред, като не се допуска многоредово представяне. Третият ред за почва със знака “+” и може да е последван от идентификатор или друга допълнителна информация. Четвъртият ред съдържа количествените характеристики, като за всяка база има точна една характеристика. Т.е. FASTQ форматът съдържа N на брой групи с информация, като N приема стойности от 1 до броя на секвенираните къси прочити.

Примерен изглед:

```
@SEQ_ID EcoliK12 4300K bp
CCTTGTCAGTAGCACTTAATCATCATGTTTAGCATTGATCTCTGCTCAATTCTT
+
!**(((***+))%%%++)(%%%%).1***-+*)***55CCF>>>>CCCCCCCC65
```

SFF - Този файлов формат е разработен от 454 и NCBI. Съдържа информация за т.нар. **Flowgram**, който представлява съвкупност от следните параметри: *base call*, *phred quality score*, *flow value*, като целта е да кодира поточните тегла на всяка нуклеотидна база, давайки мярка на вероятността базата да бъде именувана с конкретна от 4 буквената азбука {A,C,T,G}. Когато не е сигурна се приема за N. Държи се също така сметка хомополимерните участъци. Може да се използва като вход за създаване на SRA файлове.

SRA - Контейнер за данни в бинарен вид. Предимствата му са:

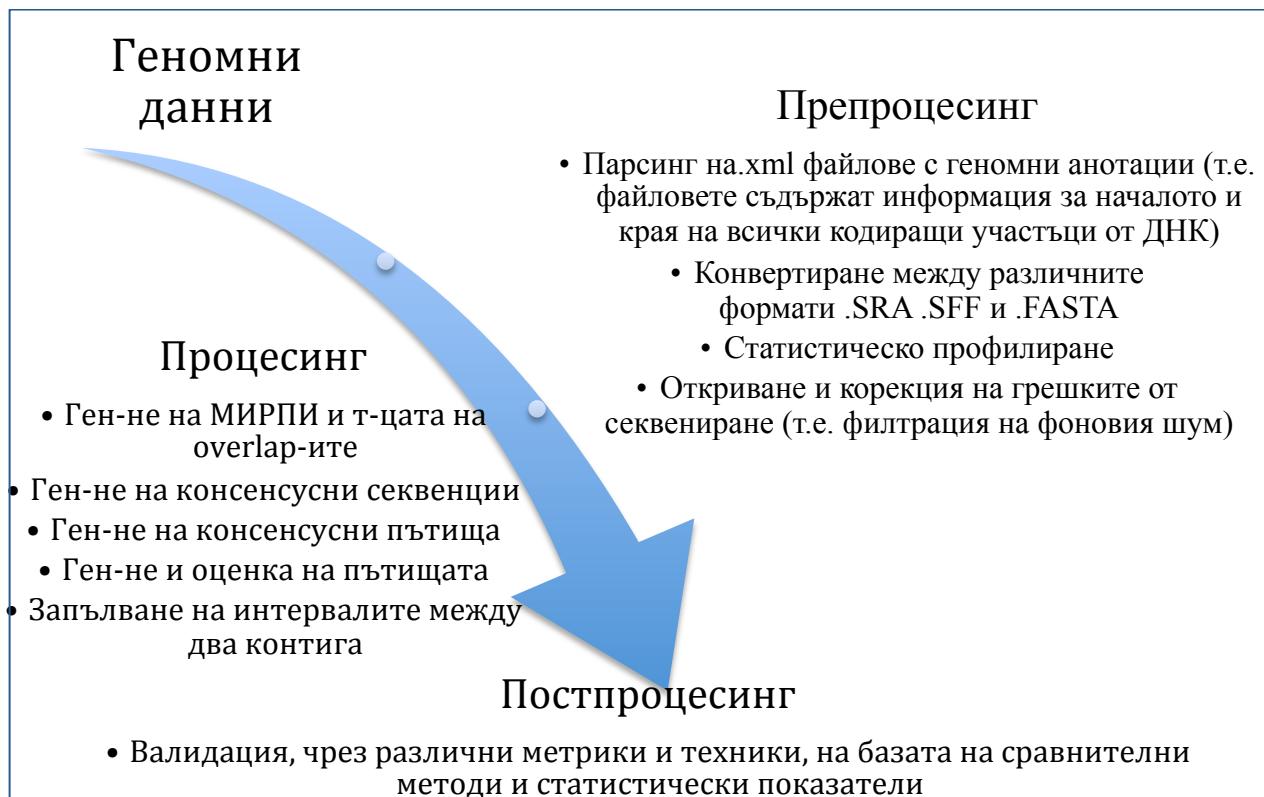
- Всички данни от едно секвениране се съдържат в един файл, като по този начин се елиминира нуждата от .tar създаване на архив.
- Данните са индексирани и позволяват случаен търсене
- Данните са в компресиран вид.
- Възможен е стрийминг, т.е. могат да бъдат четени от една входна точка

Данните са самоидентифициращи се, т.е. могат да бъдат асоциирани с името на самия файлов контейнер. Съдържат идентификатори за дата и час на секвенирането, параметри на конфигурацията и изпълнението, име на инструмента, версия и наименование на програмата, и т.н.

3. АВТОРСКИ РАЗРАБОТКИ

3.1. Обработка на данните

Използваните за анализ данни търсят следните обработки:



Фигура 13: Процеси при обработката на данни в авторската разработка

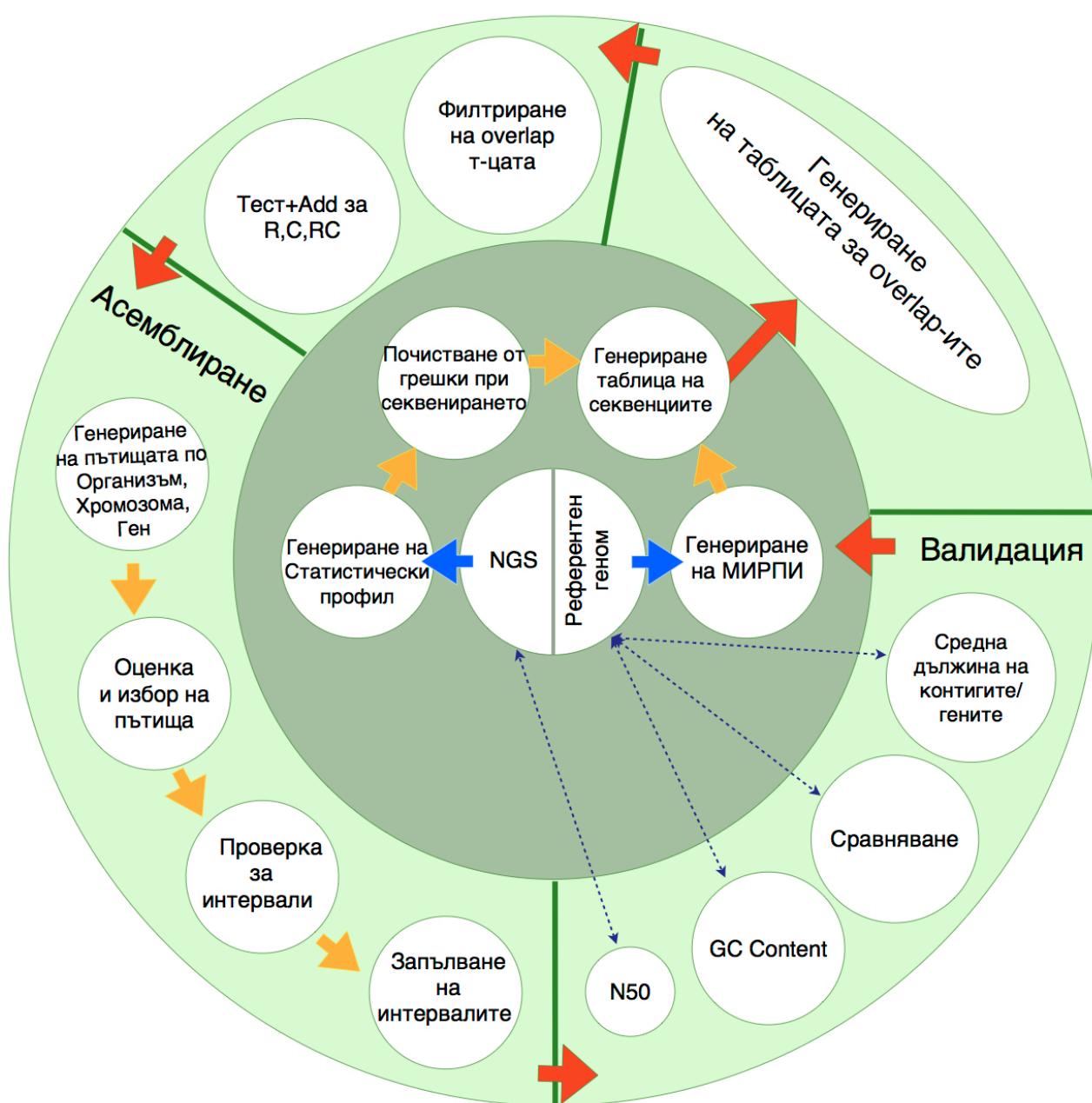
3.1. Непълнота на информацията

Данните от паралелно геномно секвениране, произведени от платформата 454 са характерни с това, че съдържат информация за всяка една секвенирана нуклеотидна база и силата на сигнала, който я е предизвикал. Има обаче случаи, в които софтуерът към платформата не може да определи със сигурност вида на базата, и в този случай записва, че е от тип N. Цел на препроцесинга е както да се изчистят данните от грешките на секвенатора, така и да се намали тази несигурност в информацията, като се прилага комплексен метод, съчетаващ статистическо профилиране и невронни мрежи за оценка и корекция.

3.2. Общ модел на алгоритъма за асемблиране, чрез използване на МИРПИ

Целият процес на асемблиране на прочитите от паралелно секвениране е представен в следната схема. Основните етапи от нея са разгледани подробно в следващите точки.

Фигура 14: Общ алгоритъм на асемблиране на къси прочити с МИРПИ

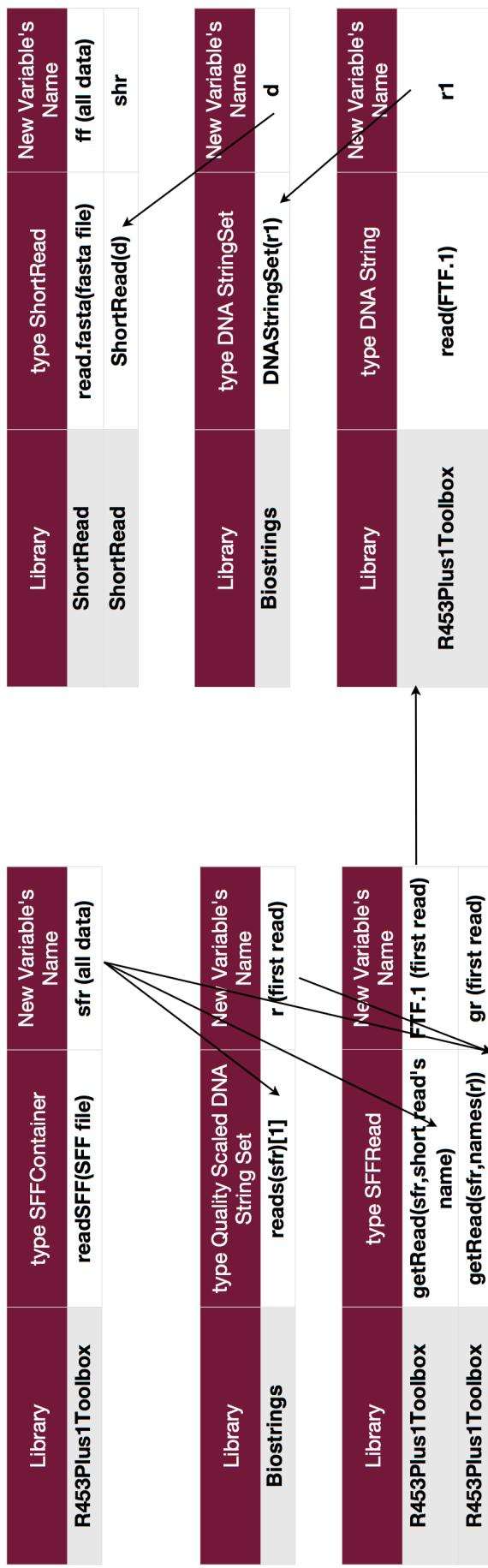


3.3. Статистическо профилиране на данните от паралелно секвениране

Разнообразието сред видовете, е значително и е главно в резултат на база концентрация, участъци от бази, и участъци от думи с необичайни честоти. Структурата на ДНК е специфична за всеки вид и се подлага само незначителни вариации по целия геном. Този геномен отпечатък може да се прилага върху данни от паралелно секвениране и разкрива тенденциите в него, т.е съставя неговия профил.

Статистическото профилиране на NGS данните е осъществено в средата на R и последващ графичен анализ в MS Excel. За първата стъпка от съществено значение е схемата на преходите за класовете данни от един вид библиотека към друг (Фигура 14).

Фигура 15: Диаграма на преходите между различните библиотеки, обработващи различни видове входни файлове в R Project

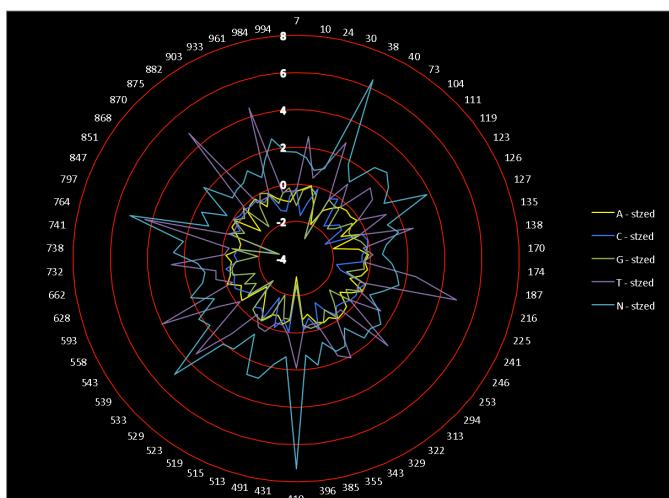


Генерирали са следните профили на данните:

- Профилиране по брой нуклеотидни бази за всеки прочит
- Профилиране по относително съдържание нуклеотидните бази за всеки прочит
- Динуклеотиден профил
- Хомополимерен профил

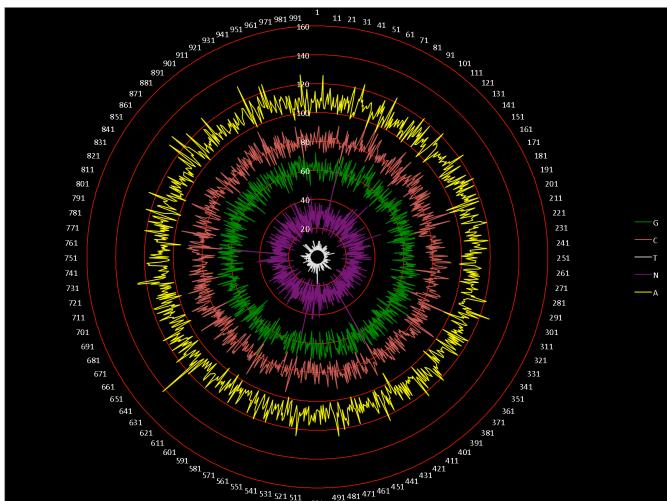
Стандартни функции от използваните библиотеки в R бяха оптимизирани за три от процеса, които значително бавеха обработката на данните. Резултатите показват около 21,69 пъти увеличаване на бързодействието, което означава, че ако 1000 къси прочита са се обработвали за около 17 минути, то след оптимизацията, общото време е около 3 минути. Това е важно, т.к. тези статистики се използват още в началото на алгоритъма за асемблиране, с цел почистване на входните данни от грешки. Освен това следва да се припомни, че броят на късите прочити е около 80 000, което би означавало, че без оптимизацията, профилирането щеше да се осъществи за 34 дни, вместо за 37 часа. Всички изчисления са проведени на машина MacBook, CPU: Intel Core2Duo, 2.4 GHz, с 4 GB RAM.

Основен извод за анализа на профилирането, е че пиковете на неопределението бази от тип N отговарят на падове за някоя от стандартните нуклеотидни бази, като не се наблюдават хомополимерни участъци на N.



Фигура 16: Ирис 1

Графика на стандартизираните величини за нуклеотидните бази в първите 1000 къси прочити. Приложен е филтър, който показва падовете на нуклеотидна база А.



Фигура 17: Ирис 2

Графика по абсолютните стойности за броя на нуклеотидните бази в първите 1000 къси прочита, без стандартизация и филтриране.

Този факт се използва за предварителна корекция на данните, след което се извършва повторно профилиране на коригираните данни. Новото профилиране се използва в невронната мрежа за отстраняване на “зашумяването” на входните данни, което е резултат както грешката при секвениране.

3.4. Откриване и филтриране на фоновия шум от NGS данни

Качеството на секвенираните данни чрез следващо поколение технологии все още се поставя под въпрос, въпреки въведените подобрения в технологичния процес на секвениране. Съществуват различни типове грешки – грешки на секвенирането, грешки на PCR, единични нуклеотидни полиморфизми (ЕНП) – на практика те не се явяват грешки, а разновидност. Общото между всички тях е, че се касае за грешно определена база – с изключение на ЕНП.

Има два основни подхода за определяне коректността на базата – множествено сравняване, придружено от тест за достоверност на сравняването, и процес, в който основните стъпки са кълстериазация на данните, сравняване и тест за достоверност.

Позовавайки се на идеята, че е възможно да се предскаже дали дадена база е с голям достоверност или не, тук се предлага един по-различен подход, използващ комплекс от методи, включващи статистически анализ и кълстериазация, приложени върху цялата съвкупност от данни, и създаване на невронна мрежа върху част от данните. Целта на невронната мрежата е чрез трениране на представителна извадка от данните да бъдат тествани останалите извън тренинг съвкупността данни.

Т.к. конкретната невронна мрежа ще съдържа данни от един и същ секвенатор, то тя ще е настроена по-скоро да открива коригира грешки, които са продукт на самия секвенатор, и в много по-малка степен би засегнала ЕНП проявленията.

Архитектурата на невронната мрежа е с универсален характер, така че може да се използва с различни NGS платформи (Sanger, Illumina, 454 и.т.), както и да се посочва типа на секвенирането – WHS, WXS, EST...

Както и в двета основни подхода за филтрация на фоновия шум, и тук ще бъде използвано сравняване на секвенции, но само върху данните, участващи в мрежата за трениране. По този начин силно се съкращава изчислителното време. Архитектурата на невронната мрежа поддържа две основни задачи:

- Откриване на “сгрешените” бази
- Предсказване на стойността на базите за откритите “сгрешени” бази

В качеството на предиктори се използват 14 параметъра – някои от тях идентификатори (за тип на организма, номер на късия прочит, номер на позията на базата, идентификатор на платформата за секвениране, идентификатор на типа секвениране). Друга част от предикторите са базирани на статистики, предварително изведени за данните, характеризиращи късия прочит. Специална група от предиктори включва информация за стойността на базата; булев израз за нейната достоверност; типа на грешката, определен в зависимост от сравняването (инсерция/делеция, или конфликт между две различни стойности за една и съща база). Последната група условно е наречена “специална”, т.к. тук са включени предикторите, които в последствие ще се използват за реализиране на двете задачи на архитектурата на невронната мрежа.

Таблица 9: Вид на трениращата таблица

Предиктор	Пояснение
SeqPlatform	Име на платформата за секвениране
SeqType	Тип на секвенирането
Org	Име на организма
SHR	Номер на късия прочит
Bn	Абсолютна позия на базата в прочита
Bv	Стойност на базата
DNProfile	Динуклеотиден профил на изходната секвенция; Динуклеотидният профил представлява функция на разпределението на динуклеотидите; стойност
Pr.1	$\text{MAX}(\text{LocalNoise}_i, \text{radius} = \pm 10)$
Pr.2	$\text{LocalNoise}_i = \text{abs}[\text{flow-round}(\text{flow})]$, където i е номера на поредната нуклеотидна база. Тук flow е коефициентът за базата от оригиналните данни. Локалният шум замерва вероятността за подценяване на оценки

	от типа $0,45 \rightarrow 0$ $(m_1 - m_0) * (s_0 + s_1)$, повишаването на Pr.3 е свързано с по-големия евентуален брой грешки, m и s са средното и стандартно отклонение, съответно за всички бази със стойност 0 или 1, т.е. този предиктор е обща характеристика на късия прочит.
Pr.3	
Pr.4	Булев предиктор, отчитащ дали конкретната база е част от хомополимер или не. Ако принадлежи = > вероятността за грешка малка.
Pr.5	Булев предиктор за отчитане на асинхрон
Pr.6	Позиция в прочита: $\text{abs}(40\text{-current base position})$
Correct	Булев “предиктор”, отразяващ дали базата е вярна или не. В зависимост от задачата може да бъде или входен или изходен неврон.
CrcType	Тип на коректността на базата според матрицата за сравняване на зашумени с N секвенции – авторска разработка. В зависимост от задачата може да бъде или входен или изходен неврон.

Трениращата таблица предполага широки възможности, т.к. допуска в тренирането да участват различни видове организми, секвенирани от различни машини с различни технологии. Колкото по-ограничено се използват нейните възможности, толкова по-спефично настроена е тя. И обратното – колкото повече от нейните възможности се използват, толкова по-универсален характер придобива. В нашия конкретен случай, данните предполагат използването на два генома от два различни организма, като единият е послужил за модификация на другия.

Поставени са следните ограничения към невронната мрежа: данните от тренинг таблицата трябва да бъдат представителна извадка на цялата съвкупност, като максималният брой на включените къси прочите е 1000. Това представлява приблизително 1/7 – ма от цялото множество.

Технология на определяне на представителната извадка:

1. Извеждане на статистики за честота на срещане на базите A,C,T,G,N – където N се явява база с несигурна стойност, т.е. не е определена като никоя от 4-те.
2. Стандартизиране на получените данни по т.1
3. Роза на ветровете върху данните от т.2
4. Изчистване на зашумяването на база съвпадение на пикове с падове между N и останалите бази – прилага се върху цялото множество от данни
5. Извеждане на статиските за хомополимерите
6. Корекция за хомополимерите на N според т.6 - прилага се върху цялото множество от данни

7. Повтаряне на стъпки: 1,2,5
8. Клъстеризация на данните от т.7

9. В зависимост от броя и процентното съотношение между клъстерите се определят групите, участващи в представителната извадка, след което на произволен принцип се избира съответното количество къси прочити от всеки клъстер.

Представителната извадка от данни преминава процедура на сравняване със софтуера RC454 към платформата 454. Резултатите плюс определените предиктори служат за конструране на таблицата за трениране, която ще разполага с около 400 000 реда с информация, т.к. всеки ред се формира от една база, а не от един къс прочит. Използва се backpropagation механизъм на обучение. Останалия сет от данни се преформатира в необходимия вид за съставяне на редовете със заявки към невронната мрежа.

Върху получени предсказани резултати се прави статистика за общия брой на откритите грешки, както и статистики за поправките и тяхната коректност. Коректността на поправките се определя чрез сравняване на поправените секвенции с референтните.

Резултатите от детекцията и изчистването на зашумяването се сравняват с RC454 върху целия сет от данни. Валидирането на резултатите най-ясно проличава в последния етап – след асемблиране на късите прочите.

Така построена методологията, позволява да се използва дори с de novo секвенционни данни, защото мрежата позволява съхраняване и трениране на данни от различни организми, но примерно от един и същ секвенатор. Това дава възможност за по-фина настройка на мрежата спрямо грешките, които се дължат на самата машина.

3.5. Изкуствени реферативни “прочити-идентификатори” (ИРПИ)

Геномът на който и да е организъм е изграден от хромозоми, съдържащи кодиращи участъци (гени / екзоны) и некодиращи участъци (интрони). Те винаги се следват: [Е]-И-Е-[И]. Базите с геномни данни поддържат информация за всеки секвениран организъм, която отразява и тези последователности от екзоны и интрони. Всъщност тази последователност може да се приеме и за своеобразен гръбнак на ДНК. Т.е. Ако разполагаме със секвенционни данни от различни технологии, но за един и същ организъм, бихме могли да се възползваме от предимството на този гръбнак, който да ни бъде опорна точка при асемблирането на данни от паралелно секвенирани данни. Въз основа на референтния геном и знанията, които сме извлечли чрез парсинг процедура за

установяване на инtron-екзон границите, съставяме изкуствени реферетни прочити-идентификатори (ИРПИ) с дължина max(дължина на късите прочити) бази, за всеки един от краишата на гените. Мрежата от ИРПИ, която създаваме ще бъде основният граф, по който асемблирането ще върви. По този начин значително се съкращава процедурата за определяне на графа, като така акцентът пада върху подграфите, т.е. от ИПРИ до ИРПИ, т.е. с оглед на данните, които обработваме, асемблираните контиги по дефиниция отговарят на гени. Поради това, че данните съдържат два генома на два различни организма, и предложената методология за асемблиране има възможност геномите да бъдат асемблиирани както по отделно, така и наведнъж.

Информацията, която се генерира въз основа на този процес по генериране на МИРПИ се записва в таблица със следния формат:

Таблица 10: Таблица на Късите прочите и ИРПИ елементите

Име на параметъра	Описание
Id_file	Името на файла, от който се извлича информацията
Id_Sequence	Идентификатор на секвенцията
Sequence_name	Име на секвенцията
Id_Chromosome	Номер на хромозата, към която принадлежи секвенцията. Прилага се когато Sequence_type=1. В останалите случаи е нула.
Sequence_type	[0;1] – т.е. може да бъде ИРПИ, когато е 1, и прочит – когато е 0
Sequence	Секвенцията, като стринг

Тази таблица е базова, и се използва при генерирането на комбинациите за overlap-ите.

3.6. Методи за генериране на консенсусни секвенции

Съставянето на консенсусна секвенция е стъпка, която се провежда след откриването на препокриващите се райони.

От гледна точка на еволюцията и мутациите съществуват следните стратегии:

- Допускаме, че има силно мутагенни фактори, стимулиращи нетипични мутации от тип пурин/пирамидин. Това допускане следва да се прилага или при третирани обекти с мутагенни факторни среди или при различни подвидове/видове.

- Допускаме, че пробите са взети от слабо мутирани източници – тогава подценяваме както мутации от типа пирамидин/пурин, така и инсерции/делеции

Много е важно да се отбележи, че не прилагаме множествено сравняване на секвенциите, т.е. при нас няма N секвенции, които се препокриват, образувайки група, а винаги само двойка секвенции – лява и дясна, като те ще могат или да се покриват в даден участък, или да се окажат еднакви. От тази гледна точка, консенсусните бази ще бъдат в пъти по-малко на брой, и следователно е оправдано прилагането на подходяща система за кодиране с буквено-цифрова оценка, на принципа на матриците, използвани при методите за сравняване. Т.к. освен 4-те бази, имаме и варианта инсерция/делеция (инсерцията в едната сравнявана секвенция се явява делеция в другата, и обратното), то кодировката е 10 цифrena (от 0 до 9). Удачно е следното кодиране:

- Съвпадащите бази не се кодират, остават в буквен вид (все пак се предполага, че районите на припокритие са с много по-малко нуклеотидни бази на брой, отколкото самите прочити).
- Зоните преди и след района на припокриване (overlap-a) не се кодират
- Консенсусната секвенция има вида:

Таблица 11: Матрица за генериране на консенсусна верига

A	G	0	Пурин/пурина	≤ 1
C	T	1	Пирамидин/Пирамидин	
A	C	3	Смесен тип, характерни мутации/грешки в секвенирането:	
A	T	5		> 1
G	C	7	Пурин/пирамидин	нечетни
G	T	9	Пирамидин/пурина	
-	A	2		
-	C	4		
-	T	6		
-	G	8		

	A	C	T	G	-
A	A	3	5	0	2
C		C	1	7	4
T			T	9	6
G				G	8

Прието е районите с припокриване да бъдат с дължина около 24 базови двойки, и да се пренебрегват двойки с подобие < 95 %. (Ye)

3.7. Конструиране на матрица и алгоритъм при “припокриване на последователности от тип overlap-и”

Когато съставяме матрицата за overlap-ите, трябва да държим сметка дали този елемент е краен, т.е. дали има само леви или десни препокривания за конкретен прочит. Тогава пътищата имат вида:

ИРПИ ₁	КП ₂	КП ₃	X1	Y1	КП ₄	КП _(k)	ИРПИ ₂
			X2	Y20			
			X6	Y35			

Т.е. един път винаги:

1. започва с ИРПИ елемент
2. завършва с ИРПИ
3. не съдържа повече от 2 ИРПИ елемента

В тази ситуация предполагаме, че можем да направим консенсусна верига, на база препокриващи се overlap-и, т.е. получаваме пътя ИРПИ₁ – ИРПИ₂. Ето един пример:

- 1) ИРПИ₁ O1 O2 O5 **O6 O8 O107**
- 2) ----- **O4 O6 O8 O109 O200 O3 O4 O7** ИРПИ₂
=====
- 3) ИРПИ₁ O1 O2 **O5 O6 O8 O109** O200 O3 O4 O7 ИРПИ₂

O1 - означава Overlap 1, т.е. сравняването е по id-тата на препокриванията. Тази методология значително ускорява процеса, т.к. по този начин избягваме множественото сравняване на секвенции, като го заместваме със сравняване на идентификатори от числов тип за вече направени сравнения по двойки секвенции, като в консенсусната верига за пътя остават тези id-та, които не са крайни, или които са с по-добра оценка.

Матрицата за припокриване, която се използва има вида:

Таблица 12: Матрица за препокриващи се региони

	A	C	T	G	N	
A	1	-3	-3	-3	1	При това положение като резултат се връщат:
C	-3	1	-3	-3	-3	<ul style="list-style-type: none"> • Начална позиция на overlap-а в секв.1 • Начална по позиция на overlap-а в секв.2 • Брой на припокритите бази • Припокритите секвенции
T	-3	-3	1	-3	1	
G	-3	-3	-3	1	-3	
N	1	-3	1	-3	1	

Тя е приложима, както при определяне на препокриващите се райони в прочитите, така и за алгоритъма за припокриване на пътищата, съдържащи крайни елементи. Присъствието на буквата N е въведено с цел допускане на сравняване на “сурови” данни, т.е. без да е извършена корекция. Тук особено важен е фактът, че при конструиране на overlap-ите се използват комбинации от 2-ри клас с n на брой елементи.

Таблица 13: Информация, генерирана от overlap-ите

Име на параметъра	Описание
Id_file	Името на файла, от който се извлича информацията
Id_overlap	Идентификатор за overlap-a
Id_pattern	Идентификатор за pattern сравняваната секвенция (Id_Sequence)
Id_subject	Идентификатор за subject сравняваната секвенция (Id_Sequence)
Pattern_name	Име на pattern секвенцията
Subject_name	Име на subject секвенцията
Pettern_type	Аналогично на Sequence type – за pattern секвенцията
Subject_type	Аналогично на Sequence type – за subject секвенцията
Start_Pattern_Position	Позиция, от която стартира припокриването в pattern секвенцията
Start_Subject_Position	Позиция, от която стартира припокриването в subject секвенцията
Score	Брой на нуклеотидните бази в препокриващия се район
Relative_Mean_Score	Средно %-то изражение на покритието (т.к. прочитите са с приблизително еднаква дължинае коректно да се използва средна величина). $\text{RelativeScore} = \frac{\text{Score}}{\text{Mean}(\text{length(pattern)}; \text{length(subject))}} * 100$
p.value	Тази статистика се измерва като се използва ttest за две вериги – по една за pattern и subject секвенциите, като всички бази, които са извън overlap-a се полагат за нули, а всички бази принадлежащи на overlap-a се полагат за единици. Колкото е по-ниска е ст-та на p.value, толкова по-добър е overlap-a, т.е събитието не случайно.
Pattern_Position	[L;R;M;Eq;NA] – т.е. ляв, десен, междуинен, еднаквост, липса на припокриване, по отношение на pattern секвенцията.
Subject_Position	Аналогично на Pattern_Position. Когато Pattern_Position е L или R, то Subject_position приема ст-сти R или L. Когато едната позиция е Eq или NA, то и другата е Eq или NA.
Overlap_sequence	Секвенцията на припокриване

За всяко изчислямо поле, което не е пряк резултат от работа на функцията за сравняване се съставя мини функция, която прилага в комбинация с *apply* конструкция.

След генерирането на последната таблица се пристъпва към детекция на повторите в данните от паралелното секвениране. Повтор е наличен когато в таблицата `Start_Pattern_Position = Start_Subject_Position`. Премахват се всички `overlap-i`, които съдържат повтори. По този начин едновременно намалява пространството за последващи обработки, без да засягаме възможните пътища, и без да елиминираме възможността за поява на повтор в друг контиг.

Следваща стъпка е къстериране на таблицата според р-стойност и определянето на различните категории препокривания. На база получените резултати, определяме категориите, които ще участват в създаването на пътищата, т.е. това ще бъде един допълнителен филтър за качество на генеририания път. Презумцията е, че колкото е по-голяма степента на покритие, толкова е по-голяма вероятността това припокриване да образува действителна секвенция.

Съществува вариант при, който няма припокритие. Тогава се преминава към прилагане на метода за запълване на интервалите в пътищата, който е развит върху концепцията «свързани краища».

3.8. Метод за оценка на пътя в графа

Определянето на пътя следва схемата:

- Прави се категоризиране на `overlap-ите` според процента на покритие: I-ви най-добри, II-ри най-добри, III-ти най-добри припокрития и т.н. Категориите са I на брой и се определят динамично след къстерилизация на резултатите от покритието.
- Избира се път с максимален коефициент на пътя. Когато са налични пътища с еднакви оценки, се предпочита този с максимален брой I-ви най-добри припокрития. Това гарантира оптимизация на самия процес на асемблиране.

Данните за всеки път, които са необходими, за да се направи избор са:

- Общ брой на върховете в пътя (подграфа)
- Идентификатор на пътя
- Брой върхове от категория I
- Брой върхове от категория II
- Брой върхове от категория III
- Брой върхове от категория...
- Брой върхове от категория I

- Общ коефициент на пътя – колкото е по-нисък коефициента, толкова по-ниско качествен е пътя, защото припокриването на секвенциите е сравнително малко.
 - Формула: $K_{id_path} = (n / \sum_{i=1}^m n_i * cat_i) * 100$
 - n – общ брой на върховете в пътя
 - i – номер на категория
 - n_i – брой на върховете в категория i
 - m – общ брой категории, $m \in [1; l]$
 - cat_i - брой точки, за категорията, $cat_i = i$
 - $K \in (0; 100]\%$

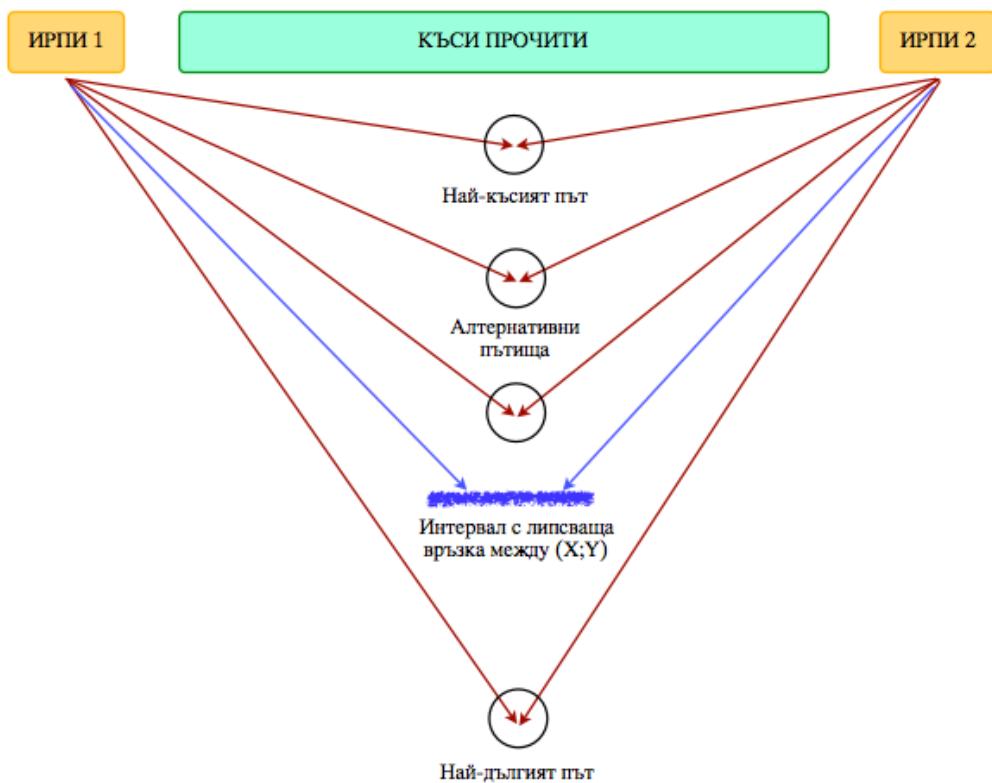
Таблица 14: Max K при различен процент на категория I в пътя

Върхове от кат.I в пътя [%]	Max K, т.e. ако всички останали са от категория II
100	100
90	91
80	83
70	77
60	71
50	67
40	63
30	59
20	56
10	53
5	51
0	50

Тази схема на оценяване на пътищата позволява провеждането на допълнително профилиране на изходните данни от паралелното секвениране под формата на статистики за средният коефициент на пътя, минимум, максимум, стандартно отклонение, тип на разпределението на данните. Тези статистики са полезни за определяне на оптималния брой възли (върхове) в графа, като тук се прави корелация със същите статистики по отношение обаче на средната дължина на гените от референтния геном.

Кога един път завършва? :

- Когато стигне до следващото ИРПИ. Основен критерий за качеството на пътя е той да минава през поредни, т.е. последователни по номер ИРПИ, в противен случай конкретният път получава флаг “discarded”, т.е. отхвърлен.
- Крайните елементи, т.е. елемент само с десен или ляв overlap, получават допълнителен флаг. Това на практика означава, ако липсва десен overlap, то след този прочит задължително се наблюдава прекъсване на пътя. Броят на тези елементи, които участват в генерирането на пътя, разделен на 2, показва броят на интервалите, т.е. на прекъсванията, които ще се наблюдават при цялостното асемблиране.



Фигура 18: Асемблиране на ИРПИ1-ИРПИ2

Както прави впечатление, алгоритъмът за асемблиране върви паралелно и насрещно, т.е. наблюдават се пътищата едновременно от посока Ляво-Дясно и Дясно – Ляво.

3.9. Метод за запълване на интервалите в пътищата

Използването на технологията «свързани краища» работи с допускането, че между всеки две секвенции, отстоящи на N на брой нуклеотидни бази една от друга, може да се намери такава последователност, която е локализирана в интервала по между им.

Този принцип използваме при съставянето на двойките $(X;Y)$, където X и Y са прочити между, които съществува интервал. Тези двойки:

- Използваме BLAST софтуер срещу референтния геном, в резултат получаваме липсваща секвенция от референтния геном.
- Спрямо получената секвенция се извършва картиране на неизползваните прочити, и се възприемат, тези, които най-добре пасват.

Тази процедура е възможно да се повтори няколко пъти за един и същ път, т.к. е вероятно ИРПИ₁ – ИРПИ₂ да бъде накъсан на повече от едно място.

Т.к. разполагаме с референтен геном и МИРПИ – построена на база

референтния геном, отпадат за разглеждане различни варианти на подреждане на секвенциите от тип ИРПИ, но остават за изясняване двойките (X;Y).

Целта е построяване на графа от ИРПИ₁ до ИРПИ₂ при следните условия:

ИРПИ ₁	X1	Y1	ИРПИ ₂
	X2	Y2	

Двойките (X1;X2) и (Y1;Y2) не се изследват, т.к. това по дефиниция са алтернативни пътища, в които X1 и X2 са крайни прочити с единствен overlap от ляво, а Y1 и Y2 са крайни прочити с единствен overlap от дясно. Ако при съпоставянето на тези двойки срещу рефентния геном не се откроят инсерции, това означава, че X1 и Y1 принадлежат на различни хромозоми. Аналогичен е изводът за X2 и Y2. В тази връзка и при съставянето на МИРПИ се взима под внимание и принадлежността към хромозомата. Тази комбинация на методи “свързани краища” и съпоставяне за принадлежност към хромозоми се приема за Greedy алгоритъм, в сравнение с чистия метод на “свързани краища”, който се счита за Brute-force алгоритъм.

ГЛАВА ЧЕТВЪРТА: РЕЗУЛТАТИ ВЪЗ ОСНОВА НА РАЗРАБОТЕНИТЕ МЕТОДИ И ПРИЛАГАНЕТО ИМ

1. Статистическо профилиране на данните от паралелно секвениране

Статистическото профилиране, както и останалите разработки претърпяха множество корекции, като в общия случай първият и последният вариант силно се отличават един от друг, както по време за изпълнение, така и понякога и по подхода за решаване. Наличието на буквата N в символния низ, отразява колебанието в силата на сигнала, което възпрепятства точното му дефиниране в границите на ДНК азбуката. От друга страна е известно, че разпределението на буквите от ДНК азбуката е един вид “пръстов отпечатък” за съответния геном. Следователно разглеждането на разпределенията в профилите на N трябва да съответства на грешката, която се генерира от машината за произвеждането на данните. Ето защо само променливите от тип: N, *N, N*, и хомополимери на N, са взети под внимание в статистическия анализ, въпреки първоначалната идея за пълно профилиране. Установи се също така че, основният пакет за обработка на .SFF¹⁷ данни - R453Plus1Toolbox не разполага със функция, която да преброява хомополимерите на N. Генерално погледнато, всички тези преброявания биха могли да се извършват с една функция и подходящи параметри, но този подход се оказа по-времеемък, отколкото съставянето на три отделни функции. В крайна сметка, получаването на статистическите профили е постигнато както следва:

Таблица 15: Време за изчисление на 3-те профила на данните по брой N, *N, N*, N^k

Операция	Време за изпълнение
Elapsed.time	5.9 mins
Reading.time	39 secs
Working.time	5.15 mins
BaseSta.time	1.45 mins
DiNuOdR.time	1.3 mins
HMPStat.time	2.4 mins
Exports.time	6 secs

За получените резултати бяха пресметнати статистическите параметри: сума, средно, стандартно отклонение, вариация, зашумяване (SNR- Signal to Noise Ratio). Последните два показатела са обратно пропорционални, като се пресмятат като отношения на средното и стандартното отклонение. Тези статистики също така бяха използвани за клъстериране. Това бе причината да се генерират графики, представлящи различните връзки вътре и между 3-те

¹⁷ т.е. освен символните низове, в него се поддържат и различна информация за силата и теглата на сигналите

профила. Те послужиха и за визуален ориентир при избора на векторите, с които се образуват точки за групиране.

Графиките, показващи профилирането са организирани по следния начин:

- Динуклеотиден профил
- Хомополимерен профил
- Комбиниран профил между динуклеотидните статистики и общия брой на N; Комбиниран профил между хомополимерните статистики и общия брой на N
- Комбиниран профил между статистиките за динуклеотидите и хомополимерите: по вертикалата са разположени хомополимерните статистики, а по хоризонтала – динуклеотидните:

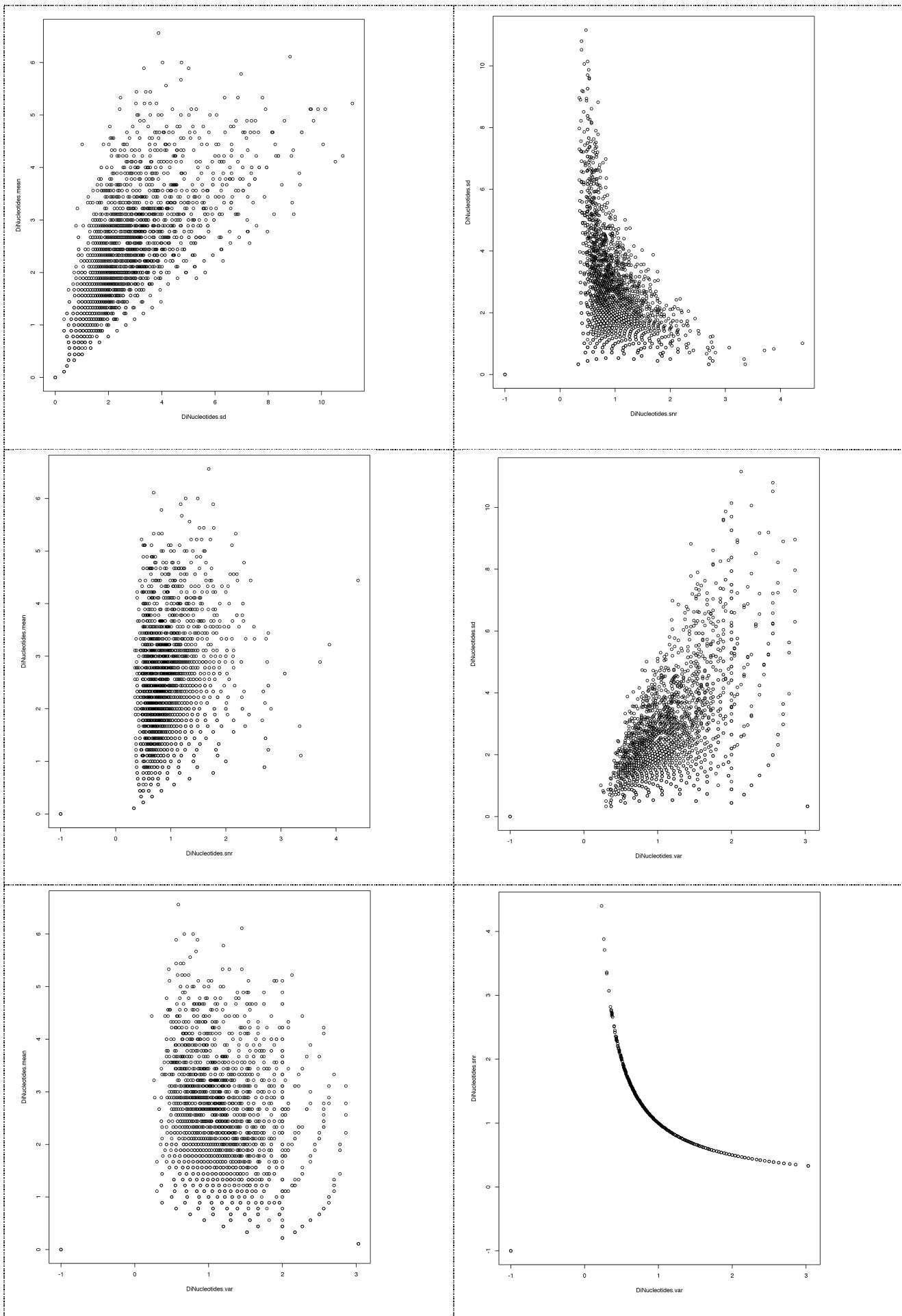
Хомополимерни статистики	Динуклетидни статистики	Средно	стандартно отклонение	SNR	Коефициент на вариация
Средно					
Стандартно отклонение					
SNR					
Коефициент на вариация					графики

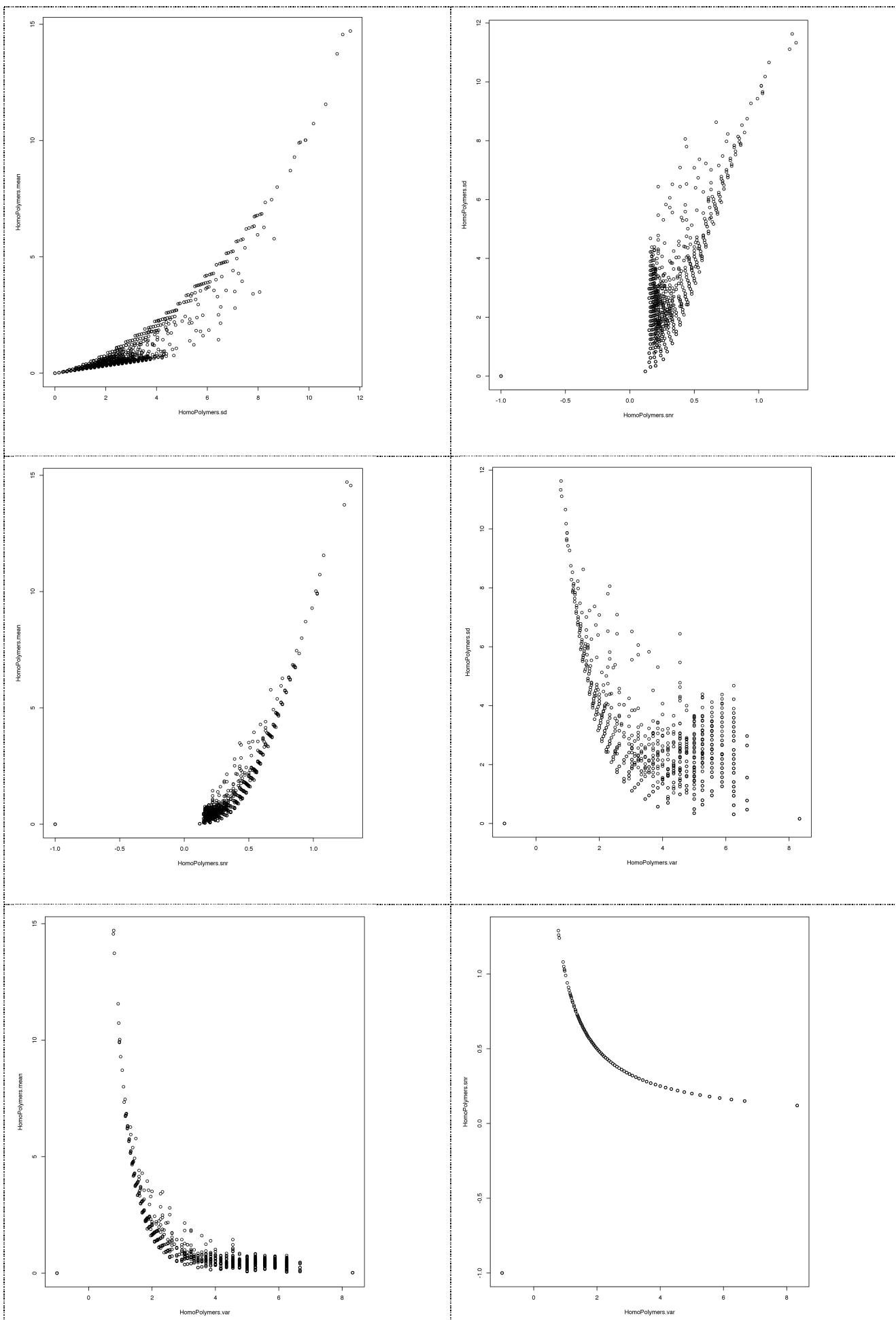
Таблица 16: Имена на графиките за Динуклеотидното профилиране, в зависимост от подредбата им

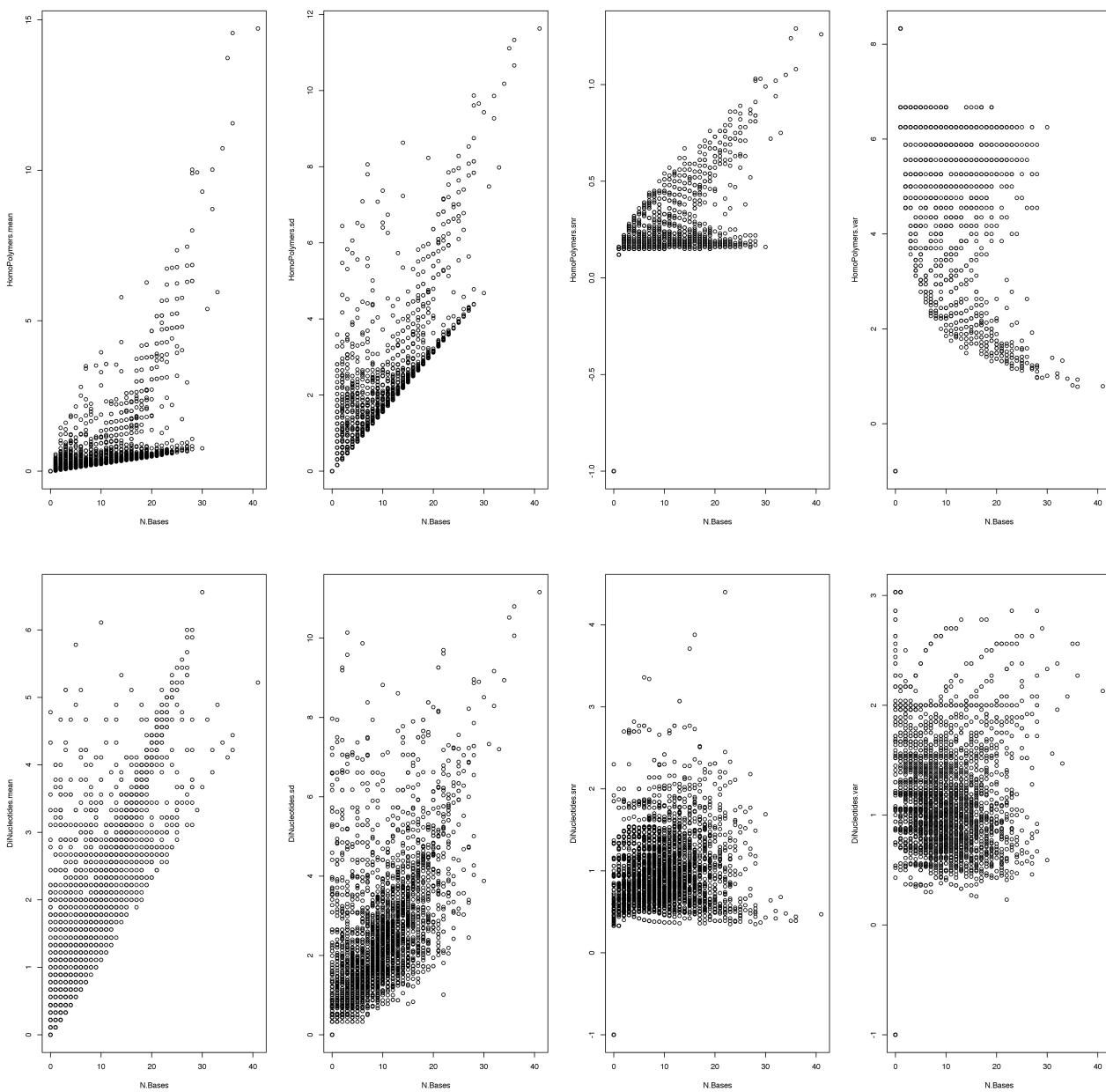
Фигура 19: Динуклеотиден профил с/д статистиките: "средно" и "стандартно отклонение"	Фигура 20: Динуклеотиден профил с/д статистиките: "стандартно отклонение" и "SNR"
Фигура 21: Динуклеотиден профил с/д статистиките: "средно" и "SNR"	Фигура 22: Динуклеотиден профил с/д статистиките: "стандартно отклонение" и "коефициент на вариация"
Фигура 23: Динуклеотиден профил с/д статистиките: "средно" и "коефициент на вариация"	Фигура 24: Динуклеотиден профил с/д статистиките: "SNR" и "коефициент на вариация"

Таблица 17: Имена на графиките за Хомополимерно профилиране, в зависимост от подредбата им

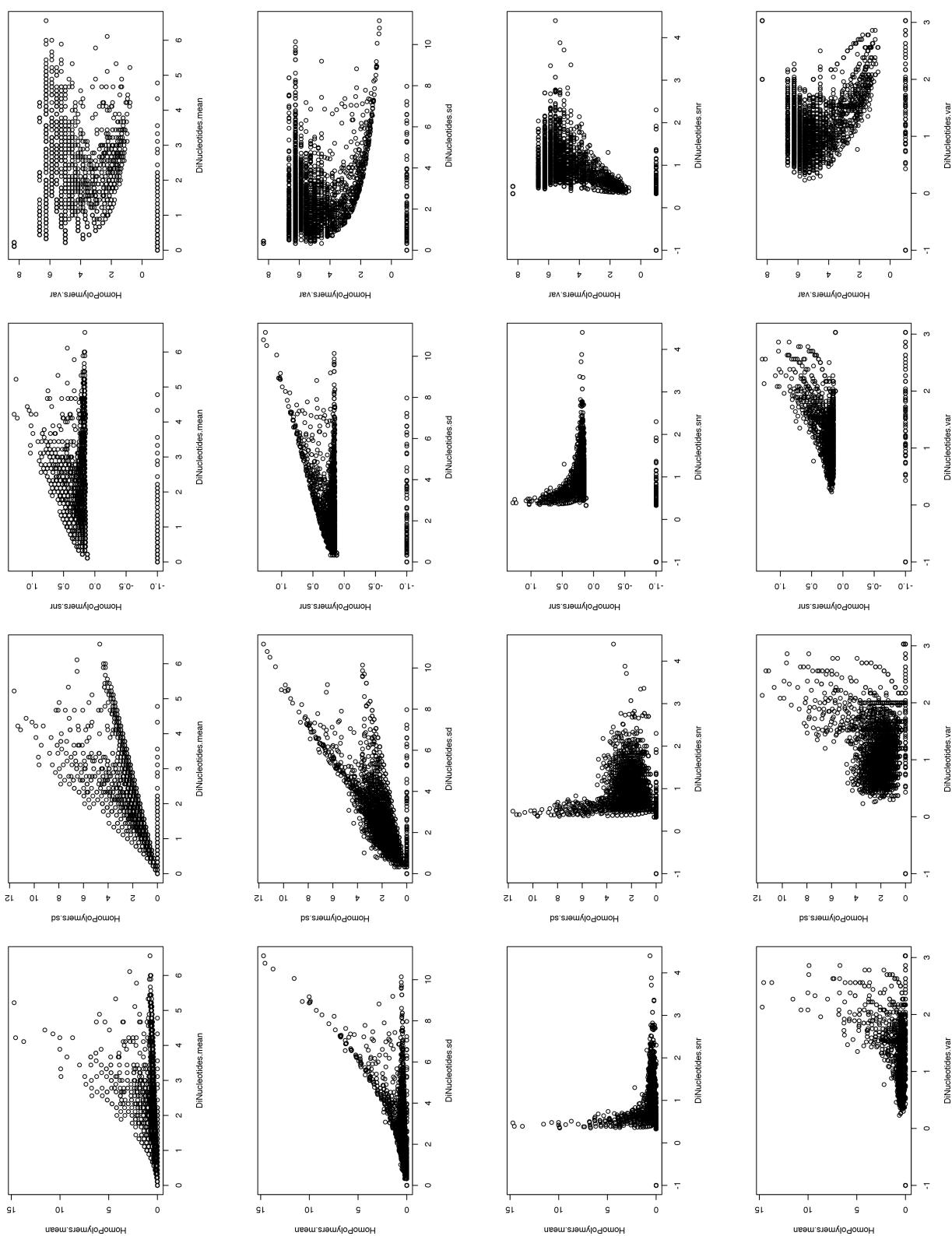
Фигура 25: Хомополимерен профил с/д статистиките "средно" и "стандартно отклонение"	Фигура 26: Хомополимерен профил с/д статистиките: "стандартно отклонени" и "SNR"
Фигура 27: Хомополимерен профил с/д статистиките: "средно" и "SNR"	Фигура 28: Хомополимерен профил с/д статистиките: "SNR" и "коефициент на вариация"
Фигура 29: Хомополимерен профил с/д статистиките: "средно" и "коефициент на вариация"	Фигура 30: Хомополимерен профил с/д статистиките: "SNR" и "коефициент на вариация"







Фигура 31: Профили между статистиките на динуклеотидите или хомополимерите (от една страна) и броя на N базите (от друга страна)



Фигура 32: Профилиране м/у динуклеотидни х-ки и хомополимерни х-ки

На базата на това графично представяне на взаимовръзките се вижда, че използването на общия брой на N може да се пренебрегне при кълстерирането. Разпределение на общия брой на N се отразява върху другите два профила, което е подтвърдено от практическите тестове за достоверност. Следователно е достатъчно да се избере вариант на статистики за динуклеотиния и хомополимерния профил, при който кълстерите ясно да си личат. Тестовете за избор на метод за кълстерилизация са разработени с използването на варианта “средното за хомополирите срещу стандартното отклонение за динуклеотидите”. Тези тестовете могат да се продължат и върху целия ред с използване на стандартно отклонение на динуклеотидите срещу статистиките за хомополирите.

При предварителните тестове за избор на тип кълстериране се установи, че най-добри (компактни) групи дава Quality Threshold метода. Бяха проведени изпитания с различен радиус и как това влияе върху разпределението на гъстотата на групите (Density Analysis). Най-удачно е използването на радиус 0.05. При тази имплементацията за пръв път в разработката възникна проблем с недостига на памет. Беше разгледан статистическият метод Bootstrap, при който данните се разделят на няколко групи, и за всяка група се пресмятат кълстерите. Особеното в този случай, е че Bootstrap се оказа приложим само за алгоритми, които използват юерархична кълстерилизация, което съществуваше в разработката като възможен вариант. Във връзка с прилагането на идеята за Bootstrap направихме опити да се използва основната библиотека на R Revolution, която е предвидена за обработка на големи количества с данни, но и там се натъкнахме на липса на подходящ инструментариум. Въпреки това, идеята за разделяне на основния пакет с данни на няколко слота се наложи, което в крайна сметка доведе до прогрес при получаването на резултати. При данни с общ обем около 80000 двумерни точки, функцията `qtclust{flexclust}` успява да обработи наведнъж 20000 за около 6.5 минути, което е лимитирано от наличната оперативна памет. По този начин функцията може да се пусне 2 пъти последователно:

1. За всеки от 4-те слота по 20000 точки. На този етап са важни точките са извън групите. Всички такива точки от 4-те слота формират една специална съвкупност, която се добавя към всеки слот.
2. Това гарантира, че всяка точка ще попадне поне в една група при повторното стартиране на функцията с така модифицираните слотове.
3. Т.к. радиусът е един същ, то се формират групи през равни интервали и е лесно резултатите да бъдат обединени.

Този метод за кълстерилизация е приложим и в други случаи при работа с големи обеми данни. Въз основа на кълстерирането става възможно определянето представителна извадка от данните, които да послужат като входни данни при реализацията на метода на изчистването на фоновия шум, произведен от самата технология за секвениране.

2. Построяване на мрежа от изкуствени реферативни “прочити - идентификатори” (МИРПИ)

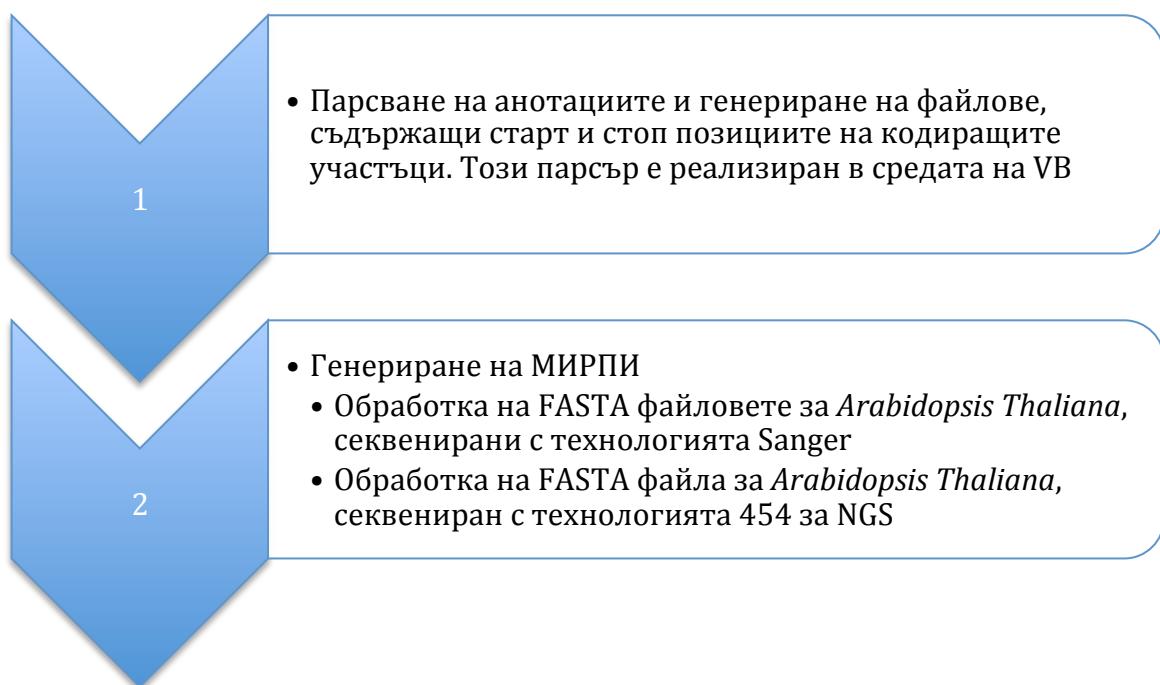
Изграждането на МИРПИ е основен етап от дисертационния труд, имащ отношение към метода за асемблиране на късите прочити от паралелно секвенирани данни.

Целта на изграждането на МИРПИ е генерирането на т.нар. “гръбнак” на граф, по който се извършват процедурите по съставяне на главния път и подпътищата от графа.

Данните, които се използват се делят на три групи:

- SRA файл → конвертиран до: .SFF и .FASTA, съдържащи секвенциите от NGS секвенирането. Т.е. съвкупността от символни низове, която подлежи на анализ и асемблиране.
- Файлове с анотации по хромозоми за генома по примера на *Arabidopsis Thaliana*, взети от базата данни на NCBI
- FASTA файлове от секвенирането по технологията Sanger, съдържащи гените по хромози, в текстов вид.

Общ вид на алгоритъма:



Вторият етап на този алгоритъм се реализира в средата на R Project и е даден под формата на функции по-долу в текста (виж т. 5.4 от Избрани реализации).

Прилагането на различни модификации на стандартни алгоритми, обработващи цялата съвкупност от данни, предизвикваха или претоварване на процесора или изчерпване на цялата памет. Резултатът и в двата случая водеше до принудителен рестарт на системата. При тест в GUI на R Revolution (специално разработена среда за големи обеми данни на базата на R), при стартиране на скрипта се предоставяше предварително информация за необходимата оперативна памет, последвано от прекъсване на изпълнение. Например за етапа “кълстериране на статическите профили” беше необходима оперативна памет от порядъка на 23-25 ГБ. Това доведе до допълнително изследване на капацитетите на различните типове променливи в R Project. Проблемът с кълстеризиацията наложи разглеждането на технология като паралелизация на процесите и bootstrapping. Стана ясно, че основните библиотеки, които се използват за обработка на геномни данни не поддържат паралелизация и поради тази причини опитите за паралелизация не дадоха резултат. От друга страна – идеята за Bootstrap във вида, в който се предлага за реализиране от библиотеката flexclust не спомогна дотолкова доколкото се приложи разделяне на данните на слотове, и прилагане на Quality Treshold с радиус 0.05 за всеки слот. Радиусът беше определен експериментално върху извадка от 1000 реда статистики за 1000 символини низа. Т.к. този подход осигурява създаването на кълстери през равни интервали, то и последващото им консолидиране не беше проблем. Същият подход за разделяне на набора от данни на слотове се приложи и на етапа за построяване на МИРПИ. Тестовете се провеждаха за асемблиране на първия ген. Това решение се взе с оглед на следните факти, като например:

Ако имаме m гена и n символни низа за асемблиране, то за да разполагаме с пълната информация за припокриващите се райони трябва да извършим $(4 \times n + m)^2 - (4 \times n + m) = (4 \times n + m) \times (4 \times n + m - 1)$. По този начин за I ген следва се извършат $4 \times n \times (4 \times n + 1) = 4 \times 79990 \times (4 \times 79990 + 1) = 319960 \times 319961 = 102.374.721.560$ сравнения. Максималният размер на таблица от типа $n \times n$ в R Project е от порядъка на около $n=30000$, значително по-малко отколкото са данните, които за обработване.

След анализ на кода на функцията за сравняване pairwiseAlignment{Biostrings} от средата на R Project се установи, че когато целим само получаване числовата стойност за оптимално сравняване на два символни низа, времето за обработка може да се съкрати поне наполовина. Така в алгоритъма се заложи филтрация на получените стойности, като се оставиха тези, при които стойността е едновременно по-голяма и по-слабо разпределена. Решението за стойността, под която да се филтратрат данните е на експериментална основа, но лесно може да автоматизира. След този процес, отново се прилага функцията pairwiseAlignment{Biostrings}, но вече не върху 79990 символни низа, а върху

приблизително 100-200 символни низа. Тази оптимизация е възможна, когато искаме функцията да върне като резултат и местата в символните низове, при които се получава припокриване, то тя всъщност извършва процес, наречен traceback, т.е. обратно проследяване, който е много по-бавен в сравнение със самото изчисление на оптималното припокриване. При тези условия, данните за сравняване на един символен низ срещу всички останали се извършва средно за около 37 минути. С цел проверка на методологията, низовото пространство би могло да се ограничи до припокривания, които могат да бъдат разделени на два класа. След това разделение може да прекъсне работата с реални данни и продължаване на тестване на методологията по асемблирането с изкуствени модели на припокриване, които формират мрежа от пътища, с цел икономия на време и компютърни ресурси.

Резултатът е Таблица 12 от Авторските разработки, която се използва на етапа по генериране на overlap-ите и конструиране на вариантите за обхождащия граф.

3. Методи за асемблиране на контигите

Представени са резултатите от съставянето на поредици от припокрити символни низове, които могат да представляват или част от ген, или цял ген (в идеалния случай). Асемблирането се осъществява по хромозоми.

Асемблирането се изразява в последователността от подпроцесите:

1. съставяне на МИРПИ,
2. генериране на консенсусни секвенции (вериги),
3. генериране на пътищата от типа ИРПИ1-ИПРИ2, където последните са елементи със Sequence_type=1 (вж. Таблица 12),
4. оценка и избор на път от тип ИРПИ1-ИПРИ2,
5. запълване на интервалите в генерираните пътища.

Получените резултати подлежат на валидация, като най-достоверен критерий се явява сравняването с референтните секвенции.

Генерирането на пътищата е свързано с обход на МИРПИ таблицата, съответно за Дясно-Ляво и Ляво-Дясно припокриване. Създава се структура от тип list, в която номерът на елемент от от този списък е номерът на формирания път, а самият елемент представлява матрица с размер $2 \times x$, където x е броят на

припокритията в МИРПИ, ред 1 е идентификаторът на припокритието, а ред 2 се изчислява в последствие след генерирането на пътищата, и представлява категорията на припоктиристието. Два основни подхода съществуват за реализацията - рекурсия и вложени цикли. Интересното е, че и тук задачата малко или много прилича на задачата за сравняване на символни низове, като тук символните низове са двойки числа, представляващи номерата на символните низове. В този случай обаче сравняването е в силно стеснено пространство и вместо припокритие по диагонал се получава припокритие в отделна точка.

На практика, припокриването на пътищата е почти същата задача като изграждането на МИРПИ, но вместо символни низове от ДНК тип разполагаме с числови низове. Двете съществени промени, които се правят са:

- Вместо използването на леви и десни краища и прилагането на локално сравняване (както е при МИРПИ), се използват целите низове и се прилага сравняване от тип “overlap”, т.е. избира се максималното припокритие между всички, в които поне единият край на който и да е от двата символни низа задължително участва в района на припокритие.
- Промяна на кодиращата таблица за съответствия и несъответствия. Тук промяната е в броя на редовете и колоните и техните наименования. Таблицата е с размер общ брой на припокритията и наименования – идентификатори на припокритията.

Припокриването на пътищата е съотносимо като консумация на време и памет спрямо алгоритъма за генериране на МИРПИ, и при равни други условия (размерност на таблицата с необходими пресмятания) ще доведе до сходни резултати, но приблизително 4 пъти по-бързо, защото ще има 4 пъти по-малко информация за запис.

4. Валидация на общия модел

4.1. Заложена валидация в основния алгоритъм

Целият алгоритъм предполага една добра основа за валидация поради следните причини:

- МИРПИ гарантира началото и края на всеки контиг
- Алгоритъмът е построен да работи паралелно и насрещно, т.е. върви се към асемблиране едновременно от ляво – на дясно, и от дясно - на ляво.
- Има както възможност фоновият шум от грешки в NGS данните предварително да бъде изчистен и резултатът да се верифицира

сравняване с референтния геном, така и да се приложи матрица за сравняване, при която N да се третира или като A или като T (т.к. статистиките показват преобладаване на падове от тип A и T при едновременно наличие на пикове на N). Въщност точно такава матрица е и приложена, т.к. са много малко символните низове, които съдържат в първите и последните 50 позиции буквата N.

- Съставен е коефициент за оценка на най-подходящият път, който е съобразен с особеностите на поставената задача.
- В модела е заложено интервалите на прекъсвания се запълват на принципа paired-ends, като се следи дали не разполагаме с по-къса секвенция, която да препокрива в средата извлечената, допълваща такава от референтния геном.

4.2. Допълнителна валидация

Поради обективни причини, задача е трудно реализуема на не сървърна работна станция. Алгоритмите, които се прилагат или биха могли да бъдат приложени се категоризират по следния начин:

Памет	Бързи	Бавни
Много	A	B
Малко	B	G

Тип B са много трудно постижими, като преобладават тип A и G, т.е. получава се едно противопоставяне между бързодействие и изискуема оперативна памет. Идеалното решение може да се постигне при използване на машини от тип сървър, с няколко процесора, много памет, големи дискови пространства и бързи алгоритми. При създаването на тестови алгоритми са използвани такива, които биват класифицирани като тип A.

Валидация на асемблирането може да се направи по няколко начина. Основният сред тях е сравняване с референтен геном (когато такъв е наличен, какъвто е и нашият случай). Други методи за валидация са определяне на грешките в GC контента, N50, сравняване с други асемблери, сравняване на дълчините на гените от референтния геном с дълчините на получените контиги от асемблирането. Грешките в GC контента са показателни за общото ниво на грешка в генома, и сравняването по този показател на асемблирания геном с референтния ни дава прилизителен отговор на въпроса с какво ниво на грешка са произведени данните от паралелното секвениране. Характерно за GC контента, е че това са динуклеотиди, намиращи се в края на интрона, които динуклеотиди са силно консервативни. Ето защо те биха могли да служат за

валидиране на резултата (т.е. да дадат приблизителен процент на грешка, инкорпориран в данните априори).

Поради ресурсната ограниченност разработените алгоритми са тествани главно в не сървърна среда.

5. Избрани реализации

5.1. VBA парсинг код за XML данни, с кодиращи/некодиращи граници

```
Dim rg As Range  
Dim oL As ListObject
```

```
Sub NCBI()  
' Annotation Data Preprocessing  
'  
    Sheets("Sheet1").Select  
    Columns("AA:AB").Select  
    Application.CutCopyMode = False  
    Selection.Copy  
    Sheets("Sheet2").Select  
    Range("A1").Select  
    ActiveSheet.Paste  
  
    ActiveSheet.ListObjects.Add(xlSrcRange, Range("$A:$B"), , xlYes).Name = _  
        "ncbi"  
    Columns("A:B").Select  
    ActiveSheet.ListObjects("ncbi").TableStyle = "TableStyleLight9"  
    ActiveSheet.Range("ncbi[#All]").RemoveDuplicates Columns:=Array(1, 2), _  
        Header:=xlYes  
    Rows("2:2").Select  
    Selection.Delete Shift:=xlUp
```

```
End Sub
```

```
Sub NCBI_2()
```

‘ Създава няколко колони от тип General, които ще съдържат формули

```
Columns("B:B").Select  
Selection.Insert Shift:=xlToRight, CopyOrigin:=xlFormatFromLeftOrAbove  
Range("ncbi[[#Headers],[Column1]]").Select  
ActiveCell.FormulaR1C1 = "Flag_Complement"
```

```
Columns("C:C").Select  
Selection.Insert Shift:=xlToRight, CopyOrigin:=xlFormatFromLeftOrAbove  
Range("ncbi[[#Headers],[Column1]]").Select  
ActiveCell.FormulaR1C1 = "Gene_Annotation"
```

```
Columns("D:D").Select  
Selection.Insert Shift:=xlToRight, CopyOrigin:=xlFormatFromLeftOrAbove  
Range("ncbi[[#Headers],[Column1]]").Select
```

```

ActiveCell.FormulaR1C1 = "Gene_Start_Stop"

Columns("E:E").Select
Selection.Insert Shift:=xlToRight, CopyOrigin:=xlFormatFromLeftOrAbove
Range("ncbi[[#Headers],[Column1]]").Select
ActiveCell.FormulaR1C1 = "ISS"

Columns("F:F").Select
Selection.Insert Shift:=xlToRight, CopyOrigin:=xlFormatFromLeftOrAbove
Range("ncbi[[#Headers],[Column1]]").Select
ActiveCell.FormulaR1C1 = "Intron_Start_Stop"

Columns("G:G").Select
Selection.Insert Shift:=xlToRight, CopyOrigin:=xlFormatFromLeftOrAbove
Range("ncbi[[#Headers],[Column1]]").Select
ActiveCell.FormulaR1C1 = "Flag_is_intron"

Columns("H:H").Select
Selection.Insert Shift:=xlToRight, CopyOrigin:=xlFormatFromLeftOrAbove
Range("ncbi[[#Headers],[Column1]]").Select
ActiveCell.FormulaR1C1 = "Gene_Number"

```

```

Columns("A:A").Select
Selection.Delete Shift:=xlToLeft

Range("A2:H2").Select
Selection.NumberFormat = "General"

```

‘От тук започва изчисляването на колоните, една от целите е създаване на колона-филтър от булев тип, която показва дали даденият участък е инtron или екзон.

```

'Flag_Complement
Range("A2").Select
ActiveCell.FormulaR1C1 =_
    "=IF(LEFT(ncbi[[#This Row],[INSDFeature_location]],1)=""c"",TRUE,FALSE)"

```

```

Range("ncbi[[#All],[Flag_Complement]]").Select
Selection.Copy
Selection.pastespecial Paste:=xlPasteValues, Operation:=xlNone, SkipBlanks _:=False, Transpose:=False

```

```

'Gene_Annotation
Range("B2").Select
ActiveCell.FormulaR1C1 =_
    "=ncbi[[#This Row],[INSDFeature_location]]"

```

```

Range("ncbi[[#All],[Gene_Annotation]]").Select
Selection.Copy
Selection.pastespecial Paste:=xlPasteValues, Operation:=xlNone, SkipBlanks _:=False, Transpose:=False

```

```
Selection.Replace What:="complement", Replacement:"", LookAt:=xlPart,
```

```

SearchOrder:=xlByRows, MatchCase:=False, SearchFormat:=False, _
ReplaceFormat:=False
Selection.Replace What:="join", Replacement:"", LookAt:=xlPart, _
SearchOrder:=xlByRows, MatchCase:=False, SearchFormat:=False, _
ReplaceFormat:=False
Selection.Replace What:"(", Replacement:"", LookAt:=xlPart, _
SearchOrder:=xlByRows, MatchCase:=False, SearchFormat:=False, _
ReplaceFormat:=False
Selection.Replace What:")", Replacement:"", LookAt:=xlPart, _
SearchOrder:=xlByRows, MatchCase:=False, SearchFormat:=False, _
ReplaceFormat:=False
Selection.Replace What:<, Replacement:"", LookAt:=xlPart, _
SearchOrder:=xlByRows, MatchCase:=False, SearchFormat:=False, _
ReplaceFormat:=False
Selection.Replace What:>, Replacement:"", LookAt:=xlPart, _
SearchOrder:=xlByRows, MatchCase:=False, SearchFormat:=False, _
ReplaceFormat:=False

'Gene_Start_Stop
Range("C2").Select
ActiveCell.FormulaR1C1 =_
"=substitute(ncbi[[#This Row],[Gene_Annotation]],"".."""",""")"

Range("ncbi[[#All],[Gene_Start_Stop]]").Select
Selection.Copy
Selection.pastespecial Paste:=xlPasteValues, Operation:=xlNone, SkipBlanks _:=False, Transpose:=False

'ISS
Range("D2").Select
ActiveCell.FormulaR1C1 =_
"=MID(ncbi[[#This Row],[Gene_Annotation]],FIND("""..""",ncbi[[#This Row],[Gene_Annotation]])+2,LEN(ncbi[[#This Row],[Gene_Annotation]])-FIND("""..""",ncbi[[#This Row],[Gene_Annotation]])-1)"

'Intron_Start_Stop
Range("E2").Select
ActiveCell.FormulaR1C1 =_
"=substitute(IFERROR(LEFT(`ncbi[[#This Row],[ISS]],LEN(ncbi[[#This Row],[ISS]])-(10-FIND("""..""",RIGHT(ncbi[[#This Row],[ISS]],10)))-1),""remove_it""),"".."""",""")"

Range("ncbi[[#All],[Intron_Start_Stop]]").Select
Selection.Copy
Selection.pastespecial Paste:=xlPasteValues, Operation:=xlNone, SkipBlanks _:=False, Transpose:=False

Range("ncbi[[#All],[ISS]]").Select
Selection.Copy
Selection.pastespecial Paste:=xlPasteValues, Operation:=xlNone, SkipBlanks _:=False, Transpose:=False

```

```

Columns("D:D").Select
Selection.Delete Shift:=xlToLeft

'Flag_is_intron
Range("E2").Select
ActiveCell.FormulaR1C1 =
"=IF(ISERROR(FIND("", "", ncbi[[#This Row],[Intron_Start_Stop]])),FALSE,TRUE)"

'Gene_Number
Range("F2").Select
ActiveCell.FormulaR1C1 = "=ROW()-1"

'Convert all formulas to values
Range("ncbi[#All]").Select
Selection.Copy
Selection.pastespecial Paste:=xlPasteValues, Operation:=xlNone, SkipBlanks _ :=False, Transpose:=False

End Sub
=====

Sub MAIN()
Call NCBI
Call del_new
Call NCBI_2
End Sub
=====

Sub del_new()

    ActiveSheet.Range("ncbi").AutoFilter Field:=1, Criteria1:=Array(_
        "CDS", "gene", "rRNA", "source", "tRNA"), Operator:=xlFilterValues
'=====>>
    Dim WB As Workbook
    Dim SH As Worksheet
    Dim Rng As Range
    Set WB = Workbooks("YourBook.xls") '<<== CHANGE
    Set SH = Sheets("Sheet2")      '<<== CHANGE
    On Error Resume Next
    Set Rng = SH.AutoFilter.Range
    Set Rng = Rng.Offset(1).Resize(Rng.Rows.Count - 1)
    Set Rng = Rng.SpecialCells(xlVisible)
    On Error GoTo 0
    If Not Rng Is Nothing Then
        Rng.Select
        Rng.Delete
    End If
    ActiveSheet.Range("ncbi").AutoFilter Field:=1
End Sub

```

5.2.Работа със SFF файлове в средата R Project

```
#####
# Working with SFF files #
#
# Used library: R453Plus1Toolbox #
#####

# Read SFF file
sfr=readSFF("SRR020799.sff")

# Get the number of SHRs
length(reads(sfr))

# Get base frequency for the whole SFF file
bf=baseFrequency(sfr)

# Get base quality stats for the whole SFF file
bqs=baseQualityStats(sfr)

# Get known short read by known read name
FTF.1=getRead(sfr, "FTFWBJ101C0JA3")

# Get the Flowgram for current short read
flowgram(FTF.1)

# Get the number of 0s flowgram
length(flowgram(FTF.1)[flowgram(FTF.1)<50])

# Get the number of 1s flowgram
length(flowgram(FTF.1)[flowgram(FTF.1)>=50 & flowgram(FTF.1)<150])

# Get the Left Clip Adapter for the first SHR
clipAdapterLeft(sfr)[1]

# Get the Right Clip Adapter for the first SHR
clipAdapterRight(sfr)[1]

# Get Left and Right Quality clip
clipQualityLeft(sfr)[1]
clipQualityRight(sfr)[1]

#Get Flow Chars for the first SHR
flowChars(sfr)[1]

#Get flow indexes for the first SHR
flowIndexes(sfr)[1]

# Get the name of the first SHR
```

```

# 1.
names(reads(sfr)[1])
# 2.
name(FTF.1)

# Get the name of the file with SHRs
name(sfr)

# Get the read by number
gr=getRead(sfr,names(reads(sfr))[1])

# Get the base frequency by the number of short read
grs=DNAStringSet(read(gr))
bfr=baseFrequency(grs)
# or
bfr=baseFrequency(DNAStringSet(read(getRead(sfr,names(reads(sfr))[1]))))

# Get the Flowgram for the fist short read
fl=flowgram(getRead(sfr,names(reads(sfr))[1]))

# Plotting dinucleotideOddsRatio:
# dinucleotideOddsRatio works with the following classes:
#   SFFContainer (f.i. sfr)
#   ShortRead    (f.i. d)
#   DNAStringSet (f.i. sh)

# Plot dinucleotideOddsRatio for the whole file
dinucleotideOddsRatio(sfr, xlab="Under-/over-representation of
dinucleotides", col="firebrick1")

# Plot dinucleotide OddsRatio for the first SHR

r1=read(gr)      # Convert SFFRead to DNAString
d=DNAStringSet(r1) # Convert DNAString to DNAStringSet
sh=ShortRead(d)   # Convert DNAStringSet to ShortRead

dinucleotideOddsRatio(d , xlab="Under-/over-representation of
dinucleotides",col="firebrick1")
dinucleotideOddsRatio(sh,      xlab="Under-/over-representation      of
dinucleotides",col="firebrick1")

```

5.3. Генериране на статистическите профили – модул с функции

```
#####
# Generating stats about NGS data and 3D filtering to get 1000 SHRs      #
#     enterprise examples for involving them in the training set          #

#
#                                     Autor: Valeriya Simeonova
#                                     CopyRight: 2014
#                                     Contact Information: v.n.simeonova@me.com
#
# FUNCTIONS' LIBRARY
#####

#----- USED LIBRARIES -----


library(R453Plus1Toolbox)
library(ShortRead)
library(seqinr)
library(timeDate)

#----- FUNCTIONS-----
# -----number of poly(di/mono)nucleotides that contains "N" base--
leNcol<-function(alphabet=s2c("acgnt"),wordsize=2){

# Function must return the number of poly(di/mono)nucleotides that
contains "N" base

ln=length(alphabet)

if (wordsize <= 1) {
    return(fc(ln,wordsize-1))
}
else {fc(ln,wordsize-1)+(ln-1)*leNcol(alphabet,wordsize-1)}

}

# ----- Recurrent self multiplication -----
fc<-function(x,n){
# Function must return x^(n-1)
# x is alphabet length
# n is wordsize
if (n<=0) return(x^n)
else x*fc(x,n-1)

}

# ----- COUNT & FREQ -----
CountAndFreq=function (FastaSeqs, StartRead=1,
StopRead=length(FastaSeqs), wordsize=1, start = 0, by = 1, count=TRUE,
freq = FALSE, alphabet = s2c("acgnt"), frame =
```

```

start,n=length(seq),OnlyN=FALSE)
{
  if (!missing(frame))
    start = frame

  OnlyN.ncol.True<-length(alphabet, wordsize=wordsize)
  OnlyN.ncol.False<-length(alphabet)^wordsize
  ncol.counts <- ifelse (OnlyN==TRUE, OnlyN.ncol.True,
  OnlyN.ncol.False)

  counts<-matrix(0,ncol= ncol.counts, nrow= StopRead-StartRead+1)

  for (k in (StartRead:StopRead)) {
    seq<-FastaSeqs[k][[1]]
    istarts <- seq(from = 1 + start, to = n, by = by)
    oligos <- seq[istarts]
    oligos.levels <- levels(as.factor(words(wordsize, alphabet =
alphabet)))
    if (wordsize >= 2) {
      for (i in 2:wordsize) {
        oligos <- paste(oligos, seq[istarts + i - 1], sep = ""))
      }
    }

    OnlyN.cond.FALSE<-table(factor(oligos, levels = oligos.levels))
    OnlyN.cond.True<-
OnlyN.cond.FALSE[grep("n",rownames(OnlyN.cond.FALSE))]
    if (OnlyN==TRUE) counts[k,1: ncol.counts] <- OnlyN.cond.True else
counts[k,1: ncol.counts] <-OnlyN.cond.FALSE
    }
    if (OnlyN==TRUE) cnts <- rownames(OnlyN.cond.True) else cnts <-
rownames(OnlyN.cond.FALSE)

    if ((count==TRUE)&(freq==TRUE)) {freqs <- counts/sum(counts);stats<-
cbind(counts,freqs);colnames(stats)=c(cnts,paste(cnts,"%"))}
    if ((count==FALSE)&(freq==FALSE)) stats="ERROR: You need counts or
freq to be set to TRUE"
    if ((count==TRUE)&(freq==FALSE)) {stats<-
counts;colnames(stats)=cnts}
    if ((count==FALSE)&(freq==TRUE)) {freqs <-
counts/sum(counts);stats<-freqs;colnames(stats)=paste(cnts,"%")}

    if (nrow(counts)>1) rownames(stats)=1:nrow(stats) else
rownames(stats)=k

    return(stats)
  }
#----Dinucleotide Odds Ratio statistics per DATA FILE by Short reads----

```

```

DNCount=function(f,n){
  # f is shr.fasta
  # shr.fasta[1][[1]] - getting the vector of nucleotides for the 1-st
  # short read
  # shr.fasta[2][[1]] - getting the vector of nucleotides for the 2-nd
  # short read
  rh.mat=matrix(0,nrow=n,ncol=9)
  for (i in (1:n)){
    rh.mat[i,1:9]=count(f[i][[1]], wordsize=2, freq = FALSE, alphabet =
      s2c("acgtn"))[c(4,9,14,16,17,18,19,20,24)]}
  rownames(rh.mat)=c(1:n)
  colnames(rh.mat)=c("AN", "CN", "GN", "NA", "NC", "NG", "NN", "NT", "TN")
  rh=as.data.frame(rh.mat)

  return(DiNu0R=rh)
}

-----Homopolymers for N bases-----
HMP=function(f,n,base.n.stats){
  b<-base.n.stats # Get Base Freqs for N
  b=as.data.frame(b)
  b1<-cbind(1:nrow(b),b) # First column is short read number
  b1<-b1[b1[,2]!=0,] # Table where n>>0
  max.n=max(b1[,2])
  a=matrix(0,ncol= max.n,nrow=n)

  for (k in(1:nrow(b1))){
    for (i in(1:b1[k,2])){
      j=b1[k,1]
      a[j,i]<-count(f[j][[1]], wordsize = i, alphabet = "n",freq=FALSE)
    }
    colnames(a)=1:max.n
    rownames(a)=1:n
  }
  return(a)
}

----- Exporting Elapsed Time per type of statistics -----
Time.tables=function(){
  time.table=list(
    Elapsed.time=time.6-time.1,
    Reading.time=time.2-time.1,
    Working.time=time.5-time.2,
    BaseSta.time=time.3-time.2,
    DiNu0dR.time=time.4-time.3,
    HMPStat.time=time.5-time.4,
    Exports.time=time.6-time.5)
  dtb=as.data.frame(time.table)
  dtb=t(dtb)
  colnames(dtb)=c("Time per operation")
  return(dtb)
}

```

5.4. Генериране на МИРПИ – модул с функции

```
#####
# MIRPI function Library #
#
# Autor: Valeriya Simeonova
# CopyRight: 2014
# Contact Information: v.n.simeonova@me.com
# FUNCTIONS' LIBRARY
#####

LoadLibs=function(){
  library(seqinr)
  library(R453Plus1Toolbox)
  library(timeDate)
}

LoadData=function(){
  #-----Setting paths and file names to FASTA files-----

  pt = "/DATA Pools/RECOVERY/WDBlack/Users/valerina/SciDev/SRR020799 -
A.Th. GMO, 454, EST strat/"
  fn.f="SRR020799.sff"
  re.f="cDNA.txt"
  sff.fasta=paste(pt,fn.f,sep="")
  ref.fasta=paste(pt,re.f,sep="")

  sff<-readSFF(sff.fasta) # Get Short Reads
  ref<-DNAString(read.fasta(ref.fasta,as.string=TRUE)[[1]][1]) # Get the
  1-st gene from 1-st Chromosome
  result=list(sff=sff,ref=ref)
  return(result)
}

#-----SHRs ranges-----
Ranges=function(n=length(reads(sff)),slots=10){
  ranges<-vector(length=slots+1)
  k=1; slot.n=n/slots
  for (i in(1:length(ranges))){ 
    ranges[i]=k
    k=k+slot.n
  }
  return(ranges=ranges)
}

# Trunc first 14 bps from every short read in Interval [k;n] in DB
GetWorkingData=function(sff,k,n){
  # Initialize reads as a list
```

```

reads<-vector("list", (n-k+1))
for (i in(k:n)){
  reads[[i]]<-sff@reads[[i]][-c(1:14)]
}
return(reads=reads)
}

# Matrix for Local Pairwise Allignment for Overlap Table Making
MAT=function(FalseValue=-
3,TrueValue=1,alphabet=c("A","C","G","T","N"),mat_size=length(alphabet))
{
mat<-matrix(FalseValue, mat_size, mat_size,dimnames=list(alphabet,
alphabet))
mat[1,1]=mat[2,2]=mat[3,3]=mat[4,4]=mat[5,5]=mat[1,5]=mat[5,1]=mat[4,5]=
mat[5,4]= TrueValue
return(mat=mat)
}

# -----Overlaps Making-----
# ----- Overlaps with Pattern = MIRPI = Left 250 bps from ref[1]

MIRPI.SOT=function(StartFromRead=1,StopOnRead=n,reads=reads,refe=data$re
f,id_ov=1,trunc2shr=250,trunc2LRsite=50,SubstitutionMatrix=mat){
# SOT is ScoreOnly=True
# ov.pat.mirpi is Left to Right local aligned, i.e. Right from RefGene
and Left from reads
ov.pat.mirpi<-matrix(0,nrow=StopOnRead-StartFromRead+1,ncol=15)
colnames(ov.pat.mirpi)<-
c("Id_Ov","Id_P","Id_S","P_Type","S_Type","Start_PPos","Start_SPos","Sco
re","RelMeanScore","pvalue","P_Pos_Type","S_Pos_Type","Overlap_PSeq","P_
Length","S_Length")
ov.pat.mirpi <-as.data.frame(ov.pat.mirpi)

ov.pat.mirpi[,2]<-1
ov.pat.mirpi[,4]<-1
di=0

seq<-refe[1+ trunc2shr-trunc2LRsite: trunc2shr]
for (i in(StartFromRead: StopOnRead)){
  di=di+1
  if(length(reads[[i]])>= trunc2LRsite){reads[[i]]=reads[[i]][1:
trunc2LRsite]}
  pa<-
pairwiseAlignment(seq,reads[[i]],substitutionMatrix=mat,type="local",sco
reOnly=TRUE)
  if (pa>3){
    ov.pat.mirpi[di,1]<-id_ov
    ov.pat.mirpi[di,3]<-i
  }
}

```

```

    ov.pat.mirpi[di,8]<-pa
    id_ov=id_ov+1}

}

ov.pat.mirpi=ov.pat.mirpi[ov.pat.mirpi[,1]!=0,]

# Overlaps with Subject = MIRPI = Righ 250 bps from ref[1]
# ov.pat.mirpi is Right to Left local aligned, i.e. Left from RefGene
and Right from reads
ov.sbj.mirpi<-matrix(0,nrow=StopOnRead-StartFromRead+1,ncol=15)
colnames(ov.sbj.mirpi)<-
c("Id_Ov","Id_P","Id_S","P_Type","S_Type","Start_PPos","Start_SPos","Sco
re","RelMeanScore","pvalue","P_Pos_Type","S_Pos_Type","Overlap_PSeq","P_
Length","S_Length")
ov.sbj.mirpi<-as.data.frame(ov.sbj.mirpi)

ov.sbj.mirpi[,3]<-1
ov.sbj.mirpi[,5]<-1
di=0

seq<-refe[(refe@length-trunc2shr +1):(refe@length-trunc2shr +
trunc2LRsite)]
for (i in(StartFromRead: StopOnRead)){
  di=di+1
  if(length(reads[[i]])>=
trunc2LRsite){reads[[i]]=reads[[i]][(reads[[i]]@length-
trunc2LRsite+1):reads[[i]]@length]}
  pa<-
pairwiseAlignment(reads[[i]],seq,substitutionMatrix=mat,type="local",sco
reOnly=TRUE)
  if (pa>3){
    ov.sbj.mirpi[di,1]<-id_ov
    ov.sbj.mirpi[di,2]<-i
    ov.sbj.mirpi[di,8]<-pa
    id_ov=id_ov+1
  }
}
ov.sbj.mirpi = ov.sbj.mirpi[ov.sbj.mirpi[,1]!=0,]
result=list(SBJ=ov.sbj.mirpi,PAT=ov.pat.mirpi,ID_OV=id_ov)
return(result)
}

MIRPI=function(MIRPI_SOT,reads=reads,refe=data$ref,trunc2shr=250,trunc2L
Rsite=50,SubstitutionMatrix=mat){
time.1=Sys.timeDate()
# ov.pat.mirpi is Left to Right local aligned, i.e. Right from RefGene
and Left from reads
ov.pat.mirpi<-MIRPI_SOT$PAT

```

```

seq<-refe[1+ trunc2shr-trunc2LRsite: trunc2shr]
for (i in (ov.pat.mirpi[,3])){
  if(length(reads[[i]])>= trunc2LRsite){reads[[i]]=reads[[i]][1:trunc2LRsite]}
  pa<-
pairwiseAlignment(seq,reads[[i]],substitutionMatrix=mat,type="local")
  ov.pat.mirpi[i,6]<-pa@pattern@range@start
  ov.pat.mirpi[i,7]<-pa@subject@range@start
  ov.pat.mirpi[i,13]<-
toString(pa@subject@unaligned[[1]][pa@subject@range@start:(pa@subject@range@start+pa@score-1)])
  ov.pat.mirpi[i,14]<-pa@pattern@unaligned@ranges@width
  ov.pat.mirpi[i,15]<-pa@subject@unaligned@ranges@width
}
# Overlaps with Subject = MIRPI = Rigth 250 bps from ref[1]
# ov.pat.mirpi is Right to Left local aligned, i.e. Left from RefGene
and Right from reads
ov.sbj.mirpi<-MIRPI_SOT$SBJ
seq<-refe[(refe@length-trunc2shr +1):(refe@length-trunc2shr +
trunc2LRsite)]
for (i in (ov.sbj.mirpi[,2])){
  if(length(reads[[i]])>=
trunc2LRsite){reads[[i]]=reads[[i]][(reads[[i]]@length-
trunc2LRsite+1):reads[[i]]@length]}
  pa<-
pairwiseAlignment(reads[[i]],seq,substitutionMatrix=mat,type="local")
  ov.sbj.mirpi[di,6]<-pa@pattern@range@start
  ov.sbj.mirpi[di,7]<-pa@subject@range@start
  ov.sbj.mirpi[di,13]<-
toString(pa@pattern@unaligned[[1]][pa@pattern@range@start:(pa@pattern@range@start+pa@score-1)])
  ov.sbj.mirpi[di,14]<-pa@pattern@unaligned@ranges@width
  ov.sbj.mirpi[di,15]<-pa@subject@unaligned@ranges@width
}
result=list(SBJ=ov.sbj.mirpi,PAT=ov.pat.mirpi,ID_OV=id_ov)
time.2=Sys.timeDate()
print(paste("Done MIRPI in ",time.2-time.1))
return(result)
}

GetSlotsStats=function(The_MIRPI,slots){
  slot_stats<-vector("list",slots)
  for (i in(1:slots)){
    slot_stats[[i]]<-
  list(table(The_MIRPI[[i]]$PAT[,8]),table(The_MIRPI[[i]]$SBJ[,8]))
  }
  return(slot_stats)
}

```

```

List2Mat=function(MyList=The_MIRPI){
n=length(MyList)
pat<-NULL
sbj<-NULL
for (i in(1:n)){
  pat<-rbind(pat,MyList[[i]]$PAT)
  sbj<-rbind(sbj,MyList[[i]]$SBJ)
}
MyMat<-list(PAT=pat,SBJ=sbj)

return(MyMat)
}

Get_MIRPI=function(r,sff.data,truncation,Lsize,id_ov,The_MIRPI,ScoreOnl
y=TRUE){
  time.1=Sys.timeDate()
  for (re in(1:(length(r)-1))){
    k=r[re];n=r[re+1]-1
    reads<-GetWorkingData(sff.data,k,n)
    The_MIRPI[[re]]<-
MIRPI.SOT(StartFromRead=k,StopOnRead=n,reads=reads,referent=data$ref,id_ov=
id_ov,trunc2shr= truncation,trunc2Lsize=Lsize,SubstitutionMatrix=mat)

    id_ov=The_MIRPI[[re]]$ID_OV
  }
  The_MIRPI<-List2Mat(The_MIRPI)
  time.2=Sys.timeDate()
  print(paste("Done MIRPI.SOT in ",time.2-time.1))
return(The_MIRPI)
}

```

ГЛАВА ПЕТА: ЗАКЛЮЧЕНИЕ

1. Дискусия на резултатите

В процеса на разработване на дисертацията, основният проблем беше изборът на данни от паралелно секвениране. Дисертацията беше замислена вместо МИРПИ да използва регуляторни елементи, но за работата с регуляторни елементи е необходимо NGS данните да бъдат от пълно геномно секвениране, т.е. да включват и кодиращите и некодиращите участъци. Такъв тип данни обаче имаше секвенирани с платформата на Illumina, но при нея късите прочити са със средна дължина 31 базови двойки, което значително щеше да усложни задачата, още повече, че тенденцията при технологиите за секвениране е генериране на данни с по-големи дължини. Затова се спряхме на данни, произведени с L454, за които е използвана EST стратегия на секвениране. Това означава, че се секвенират само кодиращите елементи от ДНК. Тези данни са характерни с това, че моделното растение *Arabidopsis Thaliana* е генно модифицирано чрез *Ti Plasmid* с линията *pBI121*. Това в началото предизвика известно смущение, но в последствие се оказа, че геномът на *pBI121* е известен, т.к. само по себе си представлява комерсиална разработка за ГМО, която цели повишаване устойчивостта на растенията срещу препарати за борба с вредителите. В този момент на избор стана ясно, че предвидените методики и алгоритми за асемблиране с помощта на регуляторни елементи (намират се в некодиращите части на генома) трябва да претърпи сериозни промени. Въз основа на тези промени беше създадена идеята за МИРПИ, като бяха променени самите методи и алгоритми. Една част от ограниченията отпаднаха, което се отрази на схемите за конструиране на пътищата, методиките за конструиране на консенсусните вериги, внесе се алгоритъм за почистване на фоновия шум на данните, генериран от грешката на секвениране, както и статистическото профилиране на прочитите. Запази се общата идея, че асемблирането на прочитите ще върви двупосочno, т.е. това са методите и алгоритмите за генериране на overlap-ите, генерирането на консенсусни пътища, оценките за последните. В първоначалния вариант бяха обмисляни две стратегии от гледна точка на бързодействие:

- Стратегия 1:
 - Генериране на пътищата чрез overlap-ите
 - Оценка на пътищата, избор

- Сдвоени краища върху резултатите срещу хромозомите
- Сравняване на инсерциите с остатъка от късите прочити
- Стратегия 2:
 - Сдвоени краища на късите прочити срещу хромозомите, с цел да се определи кой прочит към коя хромозома принадлежи, и подреждането им. Тази стратегия е полезна когато не разполагаме със знание за мрежата-гръбнак на графа, т.е. при използването на регулаторни елементи. В представеният вариант, такова дефиниране не е необходимо, т.к. МИРПИ използва информация от референтния геном, и още на етапа на генериране е известно за всеки ИРПИ елемент към коя хромозома принадлежи. След съставянето на overlap-ите, задължително се появяват такива, които да взаимодействват с ИРПИ елементите, т.е. получава се вторично разпределение на прочитите по хромозоми, т.к. всеки прочит участва в път, чието абсолютно начало е някой от ИРПИ елементите. От друга страна, много елегантно се решава и проблемът с т.нр. повтори (т.е. прочити, които се повтарят). Всички дублиращи се прочити се елиминират след генериране на overlap-ите, като се елиминират и overlap-ите, в които участват. Това не означава, че елимираме възможността даден прочит да се среща в няколко различни гена, дори от различни хромозоми – напротив. Този проблем съществуваше при вариантът с регулаторните елементи. Тогава решението беше поставянето на флаг “used”.
 - Асемблиране на наредените къси прочити, без да се налага оценка на пътя.
 - Сдвоени краища остатъците от крайни прочити срещу съответната хромозома. Неупотребените къси прочити не би следвало да има, т.к. всички са минали процес на идентификация. На този етап се интересуваме от секвенциите, за разлика от първия, където ни интересуват предимно локациите.

Беше изчислено, че за осъществяването на проверката за принадлежност на късите прочити към дадена хромозома, необходимият минимален брой операции е около 45 млрд., т.е. за целия геном на *Arabidopsis Thaliana*, това са около 225 млрд. операции. За сравнение, при стратегия 1, минимумът от операции е 45 млрд. При тези изчисления, излиза, че бихме пресмятали 1028 години принадлежността на късите прочити, ако средно една идентификация се извършва за 5 секунди.

2. Приноси

- Теоретични:
 - МИРПИ – нов подход при асемблиране на NGS данни и наличие на референтен геном
 - Използване на статистическо профилиране на NGS данните при корекция на грешките от секвенатора
 - Нов подход при избора на предиктори за конструиране на обучаващата таблица за невронната мрежа за отстраняване на грешките от секвениране
 - Нов подход за конструиране и оценка на пътищата в графа за асемблиране, като движението е паралелно и двупосочко
- Практически:
 - Оптимизация на съществуващи библиотеки в **R Project**, свързани със извеждане на статистики за данните, при първоначален тест статистики за 1000 символни низа с време за изпълнение около 3 минути, срещу постигнато време от 6.5 минути за 79990 символни низа.
 - Постигната оптимизация при сравняването на символни низове по двойки и извлечане на данни за сравняването с време за изпълнение за 79990 сравнявания \cong 37 минути, при първоначален тест за \cong 150 минути.
 - Всички тествани алгоритмични реализации са разработени, така че да може да се изпълняват върху произволна извадка от данните, както и върху цялото множество.
 - За първи път е приложен статистически анализ с използване на радиална графика («роза на ветровете») върху NGS данни.

3. Бъдещи насоки

3.1. Оптимизация на модела

Възможни са няколко вида оптимизации на действащия модел:

- Оптимизация на параметрите на асемблирация алгоритъм, в частта за генериране на пътищата, като се използват синергизми от типа Невронни мрежи- Генетичен алгоритъм.
- Оптимизация на алгоритмите по отношение на:

- Използвани структури от данни
- Съкъщаване на пространството
 - чрез Bootstrap аналогия
 - на база критерии спрямо изискванията на последващ етап (т.е. филтриране).
- Въвеждане на динамични алгоритми и рекурсия, на всички нива където това е възможно.
- Създаване на малки функции с конкретна изчислителна цел, засягаща стойността на табличен елемент. Целта е тази функция да може да се прилага чрез конструкциите `apply`, `sapply`, `tapply`, `mapply` в техния паралелизиран вид.
- Паралелизация на процесите на две нива: вътрешно (самите алгоритми), външно (според броя на ядрата на процесора, т.е. има се предвид споделена памет, а не разпределена). Според закона на Amdahl: $\text{Time}_{\text{TOTAL}} = \text{Time}_{\text{SEQUENTIAL}} + \text{Time}_{\text{PARALLEL}}$, където $\text{Time}_{\text{SEQUENTIAL}}$ е времето, за което се изпълняват непаралелизираните задачи, а $\text{Time}_{\text{PARALLEL}}$ е времето за изчисление на паралелизираните задачи. Целта е всеки алгоритъм, за който има възможност, да бъде прехвърлен в паралелната част за изчисление. Подобряването на бързината се измерва като процент спрямо времето, което е необходимо при едно ядро на процесора.

3.2. Потенциални възможности за използване на метода за асемблиране при данни от “de novo” секвениране

Алгоритъмът е приложим за всеки идентифицируем фрагмент от ДНК, който може да служи за изграждане на основния граф, по който да се извърши асемблирането. Тогава задачата за асемблиране на “de novo” геном става решима чрез разразработения метод. Тази схема ще бъде валидна при пълно геномно секвениране. Понякога новите геноми се секвенират с различни технологии. Това дава възможност за комбинирането им, като се увеличава дълбочината на секвениране, т.е. точността на асемблирането. Основният проблем в областта е липсата на достъп до нови данни. Правило е, учените, които са поръчали секвенирането на дадден геном, да го публикуват едва след като са го изследвали, което означава едно забавяне от порядъка на 3-4 години. Поради тази причина и повечето разработки се правят върху моделни организми.

4. Изводи и препоръки

Използването на МИПРИ дава голямо предимство в процеса на асемблиране, т.к. задава отправните точки, и създава т.нар “гръбнак”, по който в последствие се движи обхождането на графа. Тази техника съкръщава значително различните вероятности и силно намалява пространството на несигурност и ограничения, които се поставят от характера на данните, подлежащи на обработка.

Методологията и алгоритъмът позволяват развитие в различни посоки – от една страна при адаптация на модела е възможно прилагането му върху данни, за които не съществуват референтни геноми, от друга позволява комбинирането на данни, генериирани от различни видове платформи за секвениране, като по този начин се имитира “референтен” геном. От друга страна, в зависимост от това дали използваме МИРПИ, се появяват възможности за адаптация спрямо по-голям процент несигурност, като например ако са известни регулаторните елементи, но не и тяхната локация.

Положителна черта на предложената методология, е че позволява изпълняването на задачите поетапно – т.е. за един ген, за няколко гена, за всички гени, за една хромозома, за няколко хромозоми, за всички хромозоми. Такъв подход позволява прилагане на паралелизация на процесинга, и постигане на допълнителна оптимизация.

Ограничните ресурси са сериозна пречка, както за проверка на крайния резултат, така и за анализ на резултатите при различни параметри на алгоритмизацията.

Разработката на софтуер, който да представлява имплементация на предложния метод следва да се ръководи от:

- Характерни за средата R Project особености:
 - Избор на структури от данни от тип списък и матрица, които постепенно да бъдат запълвани вместо да се използват конструкциите за слепване `cbind()` и `rbind()`, което забавят значително процеса.
 - Задължителен предварителен преглед на програмния код на вграните функции. Прави се с цел оптимизация или корекция, в случай, че има възможност да се достъпи и друг резултат чрез промяната.
 - По възможност използване на компилирани версии на функциите, защото тогава бързодействието се увеличава.
 - Съставяне на списък от библиотеки(пакети) и функции, които позволяват паралелна обработка на данните. Установи се, че най-

широко използваните пакети като *Biostrings*, *R453Plus1Toolbox*, *ShortRead*, *seqinr* не поддържат паралелно изпълнение на функциите си за многоядрен процесор.

- Особености, свързани с прилаганите алгоритми
 - Съкръщаване на пространството
 - Въвеждане на динамични алгоритми и рекурсия, на всички нива където това е възможно.
 - Създаване на малки функции с конкретна изчислителна цел, засягаща стойността на табличен елемент. Целта е тази функция да може да се прилага чрез конструкциите *apply*, *sapply*, *tapply*, *mapply* в техния паралелизиран вид.

5. Декларация за оригиналност

Във връзка с провеждането на процедура за придобиване на образователната и научна степен "Доктор" в Софийски университет "Св. Климент Охридски" и защита на представения от мен дисертационен труд, декларирам, че резултатите и приносите на проведеното дисертационно изследване, представени в дисертационния труд на тема "Методи на soft-computing в изчислителната биология: Асемблиране на данни от геномно секвениране", са оригинални и че те не са заимствани от изследвания и публикации, в които нямам участие.

6. Публикации, участия в научни форуми и проекти, обучения.

В следващите няколко таблици се съдържа информация за проведените обучения, участията в международни научни форуми и проекти и публикациите в периода на докторантурата.

6.1.Научни публикации

Таблица 18: Научни публикации

Година	Публикация	Файл	Линк
2013	M.Krachunov, O.Kulev, V.Simeonova, M.Nisheva, D.Vassilev, 2013, <i>Fuzzy indication of reliability in metagenomic NGS data analysis</i> , IT Professional (Under review)	П1	Он-лайн
2013	V.Simeonova, K.Tasheva, G.Kosturkova, D.Vasilev. <i>A soft computing QSAR adapted model for improvement of Golden root in vitro culture growth</i> , 2013 (BBE, IF: 0.760, DOI: 10.5504/BBEQ.2013.0013	П2	Он-лайн
2012	D.Vasilev, V.Simeonova, M.Krachunov, E.Todorovska, M.Nisheva, O.Kulev, Deyan Peychev, Peter Petrov, Dimitar Shiyachki, Irena Avdjieva. <i>Data Analysis for Next Generation Sequencing – Parallel Computing Approaches in de Novo Assembly Algorithms</i> , Information Systems & Grid Technologies, Sixth International Conference ISGT'2012, Sofia, Bulgaria, June 1–3., 2012.	П3	Он-лайн
2012	V.Simeonova, A.Popov, 2012, <i>Noise discovering and correction in NGS data using artificial neural networks</i> , BIOMATH 2012 - International Conference on Mathematical Methods and Models in Biosciences and School for Young Scientists, Юни 2012, София, България	П4	Он-лайн
2012	V.Simeonova, K.Tasheva, G.Kosturkova, 2012, <i>Improvement of in vitro micropropagation efficiency of Golden Root by QSAR</i> , BIOMATH 2012 - International Conference on Mathematical Methods and Models in Biosciences and School for Young Scientists, Юни 2012, София, България	П5.1 П5.2	Он-лайн
2012	V. Simeonova, I. Popov, D.Vassilev. 2012 <i>Estimation of sequencing error rates present in genome databases</i> , BBE, DOI: 10.5504/BBEQ.2012.0070, BIOMATH 2011 Conference, Special Issue	П6	Он-лайн
2011	D.Vassilev, M.Krachunov, I.Popov, E. Todorovska, V.Simeonova, P.Szczesny, P.Siedlecki, U. Zelenkiewicz, P.Zelenkiewicz <i>Algorithm for error detection in metagenomics NGS data</i> . COST BM1006 "NGS data analysis network" (SEQAhead) meeting in Brussels - 07-09.11.2011	П7.1 П7.2	Он-лайн
2011	M. Krachunov, I. Popov, V. Simeonova, I. Avdjieva, P. Szczesny, U. Zelenkiewicz, P. Zelenkiewicz, D. Vassilev. 2011, <i>Detection and correction of errors in metagenomic 16S RNA parallel sequences</i> , The third StatSeq workshop on "Statistical challenges on the 1000€ genome sequences in plants": 14-15 April 2011 - Toulouse, France - oral	П8	Он-лайн

	presentation		
2011	M.Krachunov, I.Popov, P.Petrov, V.Simeonova, M.Nisheva, E.Todorovska, D.Vassilev, 2011, <i>Data analysis of next generation sequencing metagenomics studies - parallel computing approaches in genome assembly algorithms</i> , The 2nd Regional Conference “Supercomputing Applications in Science and Industry”: 20-21 September 2011 – Synny Beach, Bulgaria	П9	Он-лайн
2010	V.Simeonova, I.Popov, D.Vasilev, 2010, <i>Using intron-exon boundaries as a way to measure the sequencing errors in Oryza sativa genome</i> – oral presentation, workshop Bioinformatics meeting - “Bioinformatics and miRNAs”, VARNA, JUNE 2010	П10	Он-лайн
2010	Т.Балабанов, И.Занкински, В.Симеонова, 2010, "Прогнозиране на времеви редове с изкуствени невронни мрежи и диференциална еволюция в разпределена среда", Работни статии на ИИТ, ПТ/WP-268B, Май 2010	П11	Он-лайн
2010	I.Popov, V.Simeonova, D.Vasilev "Estimation of sequencing error rates present in database sequences of the <i>Oriza Sativa(Japonica Cultivar group) genome</i> " - 2nd workshop of EU COST action TD0801 Statistical challenges on the 1000€ genome sequences in plants (StatSEQ), Het Pand, Univ. of Ghent, Ghent, Belgium, May 20-21, 2010., p.34.	П12	Он-лайн
2010	T.Balabanov, I.Zankinski, V.Simeonova, 2010, "Forecasting time series with artificial neural networks and differential evolution in distributed environment ", Working papers of IIT/WP-268B, MAY 2010	П13	
2008	В.Симеонова, <i>Възможностите на ms office 2007 в теоретично обяснени и решени задачи с приложени адаптации към ms office 2003</i> , Стено, 2008, Варна, ISBN:978-954-449-392-9	П14	Он-лайн
2008	Т.Атанасова, В.Симеонова, 2008, <i>Възможности за прогнозиране на борсови индекси с невронни мрежи и генетични алгоритми</i> - Science-Practical Conference “Firms and Markets in Bulgaria in terms of European integration - the continuing adaptation, 15-16.05.2008	П15	Он-лайн

6.2. Участия в научни форуми и проекти

Таблица 19: Участия в научни форуми и проекти

Година	Форум*
2012	Участие в международна конференция BIOMATH 2012 - International Conference on Mathematical Methods and Models in Biosciences and School for Young Scientists, Юни 2012, София, България
2012	Участие в международна конференция Information Systems & Grid Technologies 2012 - Sixth International Conference ISGT, Юни 2012, София, България
2011	Участие в международна конференция BIOMATH 2011 - International Conference on Mathematical Methods and Models in Biosciences and School for Young Scientists, Юни 2011, София, България
2011	Участие в съвместен проект със ИФРГ и ИОХЦФ към БАН “Приложение на биотехнологични и биоинформатични методи за получаване <i>in vitro</i> на биомаса от застрашения вид <i>Rhodiola rosea</i> (златен корен) и за повишаване продукцията на биологично активни вещества”, спечелил финансиране по програма на Фонд “Научни изследвания” към МОНМ: НАЦИОНАЛЕН КОНКУРС „МЛАДИ УЧЕНИ – 2011 г.”.
2011	The 2nd Regional Conference “Supercomputing Applications in Science and Industry”: 20-21 September 2011 – Sunny Beach, Bulgaria
2011	The 3th workshop of EU COST action TD0801 Statistical challenges on the 1000€ genome sequences in plants (StatSEQ), Toulouse France, April 14-15.
2011	MEETINGS OF EU COST ACTION BM 1006: SEQAHEAD, 07.-09. November.2011: MC Business Meeting and Scientific Meeting @ EU COST Office, Brussels, Belgium
2010	In Proceedings of FP7 COST Action Statistical Sequencing of Plant Genome, 2nd workshop, University of Ghent, Belgium, 20-21 May 2010*
2010	Участие в конференция Bioinformatics meeting - “Bioinformatics and miRNAs”, Юни 2010, Варна, България

6.3. Участия в семинари и учебен процес по време на докторантурата

Таблица 20: Участия в семинари и учебен процес по време на докторантурата

Година	
2009-2013	Регулярен семинар по биоинформатика към Агробио институт и Геномен център към СУ „Климент Охридски”
2010-2012	Асистент по дисциплината “Приложен Софтуер”, ФМИ, СУ „Климент Охридски”
2011-2012	Практикум “Офис технологии”, ФМИ, СУ „Климент Охридски”
2013	Практикум “Езици R и Bioconductor”, Биологически факултет към Пловдивски университет “Паисий Хиландарски”, магистърска програма “Биоинформатика”

* Линкът води към интернет сайт на мероприятиято

** Линкът е към сертификата

6.1.Обучения по време на докторантурата

Таблица 21: Обучения по време на докторантурата

Година	Държава	Организация / Лектор	Наименование на курса
2009	България, САЩ	лектор д-р Иван Иванов - професор към Texas A & M University, USA	Advanced computational biology and bioinformatics
2009	САЩ, онлайн	Bioinformatics.org	Сертифициран уеб курс: R for Biologists - Level 1 and Level 2 – сертификати за ниво 1** и 2**
21-26 Юни 2010	Триест , Италия	ICGEB	Bioinformatics: Computer Methods in Molecular Biology**
13-17 Декември 2010	Атина, Гърция	COST Action FA0604	Bioinformatics in Plant Sciences**
03-07 Октомври 2011	Триест , Италия	ICGEB	Advanced Bioinformatics: Computational Systems Biology**

6.2.Цитирания

Доколкото ни е известно, към този момент изброените публикации по същността на настоящата дисертация нямат цитирания от други автори, което вероятно се дължи в някаква степен на обстоятелството, че тези публикации станаха факт едва в рамките на последните 12 - 15 месеца преди защитата на дисертационния труд.

6.3.Благодарности

За достигането ми от февруари 2010 г., когато бях зачислена като редовен докторант във ФМИ, до 2013 г. и до този финален етап, предшестващ защитата на настоящата дисертация, дължа благодарности на много хора.

Наи \square -големи благодарности дължа на моят научен ръководител доц. д-р Димитър Василев, на проф. Красен Стефанов, който прие да ми стане научен ръководител след кончината на доц. д-р Антоний Попов, както и на родителите ми, които ме подкрепят безрезервно и вярват в мен.

Благодаря още на колегите си докторанти Милко Крачунов и Огнян Кулев, както и на Светла Горанова, завеждащ докторантурите при ФМИ.

ИЗПОЛЗВАНА ЛИТЕРАТУРА

- [1] X. Li, D. Ruan, and A. J. van der Wal, “Discussion on soft computing at FLINS’96,” *Int. J. Intell. Syst.*, vol. 13, no. 2–3, pp. 287–300, 1998.
- [2] D. Gonze, “Analyse de séquences macromoléculaires: PAM and BLOSUM substitution matrices,” *Teaching activities: Course “INFO-F-434 - Analyse de séquences macromoléculaires,”* 2013. [Online]. Available: http://student.ulb.ac.be/~dgonze/TEACHING/pam_blosum.pdf. [Accessed: 14-Feb-2014].
- [3] CLC, “Use of scoring matrices.” [Online]. Available: http://www.clcsupport.com/clcgenomicsworkbench/650/Use_scoring_matrices.html. [Accessed: 14-Feb-2014].
- [4] S. R. Eddy, “Where did the BLOSUM62 alignment score matrix come from?,” *Nat. Biotechnol.*, vol. 22, no. 8, pp. 1035–1036, 2004.
- [5] S. Vinga and J. Almeida, “Alignment-free sequence comparison - a review,” *Bioinformatics*, vol. 19, pp. 513–523, 2003.
- [6] S. B. Needleman and C. D. Wunsch, “A general method applicable to the search for similarities in the amino acid sequence of two proteins,” *J. Mol. Biol.*, vol. 48, no. 3, pp. 443–453, Mar. 1970.
- [7] T. F. Smith and M. S. Waterman, “Identification of common molecular subsequences.,” *J. Mol. Biol.*, vol. 147, no. 1, pp. 195–7, Mar. 1981.
- [8] C. M. Henneke, “A multiple sequence alignment algorithm for homologous proteins using secondary structure information and optionally keying alignments to functionally important sites.,” *Comput. Appl. Biosci.*, vol. 5, no. 2, pp. 141–50, Apr. 1989.
- [9] A. M. Phillippy, M. C. Schatz, and M. Pop, “Genome assembly forensics: finding the elusive mis-assembly.,” *Genome Biol.*, vol. 9, no. 3, p. R55, 2008.
- [10] Illumina, “Whole-Genome Sequencing | Illumina.” [Online]. Available: http://www.illumina.com/applications/sequencing/dna_sequencing/whole_genome_sequencing.ilmn. [Accessed: 01-Apr-2014].
- [11] J. Kececioglu and J. Yu, “Separating repeats in DNA sequence assembly,” in *Proceedings of the fifth annual international conference on Computational biology*, 2001, pp. 176–183.
- [12] N. Whiteford, N. Haslam, G. Weber, A. Prügel-Bennett, J. W. Essex, P. L. Roach, M. Bradley, and C. Neylon, “An analysis of the feasibility of short read sequencing.,” *Nucleic Acids Res.*, vol. 33, no. 19, p. e171, 2005.
- [13] E. W. Myers, “Toward simplifying and accurately formulating fragment assembly.,” *J. Comput. Biol.*, vol. 2, no. 2, pp. 275–290, 1995.

- [14] R. M. Idury and M. S. Waterman, “A new algorithm for DNA sequence assembly.,” *J. Comput. Biol.*, vol. 2, no. 2, pp. 291–306, 1995.
- [15] D. R. Zerbino and E. Birney, “Velvet: algorithms for de novo short read assembly using de Bruijn graphs.,” *Genome Res.*, vol. 18, no. 5, pp. 821–829, 2008.
- [16] P. A. Pevzner, H. Tang, and G. Tesler, “De novo repeat classification and fragment assembly.,” *Genome Res.*, vol. 14, no. 9, pp. 1786–1796, 2004.
- [17] D. Zhi, B. J. Raphael, A. L. Price, H. Tang, and P. A. Pevzner, “Identifying repeat domains in large genomes.,” *Genome Biol.*, vol. 7, no. 1, p. R7, 2006.
- [18] D. Fasulo, A. Halpern, I. Dew, and C. Mobarry, “Efficiently detecting polymorphisms during the fragment assembly process.,” *Bioinformatics*, vol. 18 Suppl 1, pp. S294–S302, 2002.
- [19] N. Nagarajan and M. Pop, “Parametric complexity of sequence assembly: theory and applications to next generation sequencing.,” *J. Comput. Biol.*, vol. 16, no. 7, pp. 897–908, 2009.
- [20] M. Pop, “Genome assembly reborn: recent computational challenges.,” *Brief. Bioinform.*, vol. 10, no. 4, pp. 354–366, 2009.
- [21] M. Pop and S. L. Salzberg, “Bioinformatics challenges of new sequencing technology.,” *Trends Genet.*, vol. 24, no. 3, pp. 142–149, 2008.
- [22] R. L. Warren, G. G. Sutton, S. J. M. Jones, and R. A. Holt, “Assembling millions of short DNA sequences using SSAKE,” *Bioinformatics*, vol. 23, no. 4, pp. 500–501, 2007.
- [23] R. L. Warren and R. A. Holt, “SSAKE 3.0: Improved speed, accuracy and contiguity,” in *Pacific Symposium on Biocomputing*, 2008, p. 570.
- [24] J. C. Dohm, C. Lottaz, T. Borodina, and H. Himmelbauer, “SHARCGS, a fast and highly accurate short-read assembly algorithm for de novo genomic sequencing.,” *Genome Res.*, vol. 17, no. 11, pp. 1697–1706, 2007.
- [25] W. R. Jeck, J. A. Reinhardt, D. A. Baltrus, M. T. Hickenbotham, V. Magrini, E. R. Mardis, J. L. Dangl, and C. D. Jones, “Extending assembly of short DNA sequences to handle error.,” *Bioinformatics*, vol. 23, no. 21, pp. 2942–2944, 2007.
- [26] J. A. Reinhardt, D. A. Baltrus, M. T. Nishimura, W. R. Jeck, C. D. Jones, and J. L. Dangl, “De novo assembly using low-coverage short read sequence data from the rice pathogen *Pseudomonas syringae* pv. *oryzae*.,” *Genome Res.*, vol. 19, no. 2, pp. 294–305, 2009.
- [27] S. M. D. Goldberg, J. Johnson, D. Busam, T. Feldblyum, S. Ferriera, R. Friedman, A. Halpern, H. Khouri, S. A. Kravitz, F. M. Lauro, K. Li, Y.-H. Rogers, R. Strausberg, G. Sutton, L. Tallon, T. Thomas, E. Venter, M. Frazier, and J. C. Venter, “A Sanger/pyrosequencing hybrid approach for the generation of high-quality draft assemblies of marine microbial genomes.,” *Proc. Natl. Acad. Sci. U. S. A.*, vol. 103, no. 30, pp. 11240–11245, 2006.
- [28] E. W. Myers, G. G. Sutton, A. L. Delcher, I. M. Dew, D. P. Fasulo, M. J. Flanigan, S. A. Kravitz, C. M. Mobarry, K. H. Reinert, K. A. Remington, E. L. Anson, R. A. Bolanos, H. H.

- Chou, C. M. Jordan, A. L. Halpern, S. Lonardi, E. M. Beasley, R. C. Brandon, L. Chen, P. J. Dunn, Z. Lai, Y. Liang, D. R. Nusskern, M. Zhan, Q. Zhang, X. Zheng, G. M. Rubin, M. D. Adams, and J. C. Venter, “A whole-genome assembly of *Drosophila*.,” *Science*, vol. 287, no. 5461, pp. 2196–2204, 2000.
- [29] S. Batzoglou, D. B. Jaffe, K. Stanley, J. Butler, S. Gnerre, E. Mauceli, B. Berger, J. P. Mesirov, and E. S. Lander, “ARACHNE: a whole-genome shotgun assembler.,” *Genome Res.*, vol. 12, no. 1, pp. 177–189, 2002.
- [30] D. B. Jaffe, J. Butler, S. Gnerre, E. Mauceli, K. Lindblad-Toh, J. P. Mesirov, M. C. Zody, and E. S. Lander, “Whole-genome sequence assembly for mammalian genomes: Arachne 2.,” *Genome Res.*, vol. 13, no. 1, pp. 91–96, 2003.
- [31] X. Huang and S.-P. Yang, “Generating a genome assembly with PCAP.,” *Curr. Protoc. Bioinformatics*, vol. Chapter 11, p. Unit11.3, 2005.
- [32] S. Batzoglou, “Algorithmic Challenges in Mammalian Genome Sequence Assembly,” in *Encyclopedia of genomics, proteomics and bioinformatics*, M. Dunn, L. Jorde, P. Little, and S. Subramaniam, Eds. Hoboken (New Jersey): John Wiley and Sons, 2005.
- [33] M. Pop, “DNA sequence assembly algorithms,” in *Yearbook of Science and Technology*, McGraw-Hill, Ed. McGraw-Hill, 2006.
- [34] G. Sutton and I. Dew, “Shotgun Fragment Assembly,” in *Systems Biology: Genomics*, I. Rigoutsos and G. Stephanopoulos, Eds. New York: Oxford University Press, 2007, pp. 79–117.
- [35] J. R. Miller, A. L. Delcher, S. Koren, E. Venter, B. P. Walenz, A. Brownley, J. Johnson, K. Li, C. Mobarry, and G. Sutton, “Aggressive assembly of pyrosequencing reads with mates.,” *Bioinformatics*, vol. 24, no. 24, pp. 2818–2824, 2008.
- [36] D. Hernandez, P. François, L. Farinelli, M. Osterås, and J. Schrenzel, “De novo bacterial genome sequencing: millions of very short reads assembled on a desktop computer.,” *Genome Res.*, vol. 18, no. 5, pp. 802–809, 2008.
- [37] M. S. Hossain, N. Azimi, and S. Skiena, “Crystallizing short-read assemblies around seeds.,” *BMC Bioinformatics*, vol. 10 Suppl 1, p. S16, 2009.
- [38] P. A. Pevzner, “1-Tuple DNA sequencing: computer analysis.,” *J. Biomol. Struct. Dyn.*, vol. 7, no. 1, pp. 63–73, 1989.
- [39] P. A. Pevzner, H. Tang, and M. S. Waterman, “An Eulerian path approach to DNA fragment assembly.,” *Proc. Natl. Acad. Sci. U. S. A.*, vol. 98, no. 17, pp. 9748–9753, 2001.
- [40] J. T. Simpson, K. Wong, S. D. Jackman, J. E. Schein, S. J. M. Jones, and I. Birol, “ABySS: a parallel assembler for short read sequence data.,” *Genome Res.*, vol. 19, no. 6, pp. 1117–1123, 2009.
- [41] P. A. Pevzner and H. Tang, “Fragment assembly with double-barreled data.,” *Bioinformatics*, vol. 17 Suppl 1, pp. S225–S233, 2001.

- [42] M. Chaisson, P. Pevzner, and H. Tang, “Fragment assembly with short reads.,,” *Bioinformatics*, vol. 20, no. 13, pp. 2067–2074, 2004.
- [43] M. J. Chaisson and P. A. Pevzner, “Short read fragment assembly of bacterial genomes.,,” *Genome Res.*, vol. 18, no. 2, pp. 324–330, 2008.
- [44] M. J. Chaisson, D. Brinza, and P. A. Pevzner, “De novo fragment assembly with short mate-paired reads: Does the read length matter?,,” *Genome Res.*, vol. 19, no. 2, pp. 336–346, 2009.
- [45] J. C. Dohm, C. Lottaz, T. Borodina, and H. Himmelbauer, “Substantial biases in ultra-short read data sets from high-throughput DNA sequencing.,,” *Nucleic Acids Res.*, vol. 36, no. 16, p. e105, 2008.
- [46] D. R. Zerbino, G. K. McEwen, E. H. Margulies, and E. Birney, “Pebble and rock band: Heuristic resolution of repeats and scaffolding in the velvet short-read de novo assembler,” *PLoS One*, vol. 4, no. 12, 2009.
- [47] “SOLiD.” .
- [48] J. Butler, I. MacCallum, M. Kleber, I. A. Shlyakhter, M. K. Belmonte, E. S. Lander, C. Nusbaum, and D. B. Jaffe, “ALLPATHS: de novo assembly of whole-genome shotgun microreads.,,” *Genome Res.*, vol. 18, no. 5, pp. 810–820, 2008.
- [49] I. Maccallum, D. Przybylski, S. Gnerre, J. Burton, I. Shlyakhter, A. Gnirke, J. Malek, K. McKernan, S. Ranade, T. P. Shea, L. Williams, S. Young, C. Nusbaum, and D. B. Jaffe, “ALLPATHS 2: small genomes assembled accurately and with high continuity from short paired reads.,,” *Genome Biol.*, vol. 10, no. 10, p. R103, 2009.
- [50] R. Li, H. Zhu, J. Ruan, W. Qian, X. Fang, Z. Shi, Y. Li, S. Li, G. Shan, K. Kristiansen, S. Li, H. Yang, J. Wang, and J. Wang, “De novo assembly of human genomes with massively parallel short read sequencing.,,” *Genome Res.*, vol. 20, no. 2, pp. 265–272, 2010.
- [51] R. Li, W. Fan, G. Tian, H. Zhu, L. He, J. Cai, Q. Huang, Q. Cai, B. Li, Y. Bai, Z. Zhang, Y. Zhang, W. Wang, J. Li, F. Wei, H. Li, M. Jian, J. Li, Z. Zhang, R. Nielsen, D. Li, W. Gu, Z. Yang, Z. Xuan, O. A. Ryder, F. C.-C. Leung, Y. Zhou, J. Cao, X. Sun, Y. Fu, X. Fang, X. Guo, B. Wang, R. Hou, F. Shen, B. Mu, P. Ni, R. Lin, W. Qian, G. Wang, C. Yu, W. Nie, J. Wang, Z. Wu, H. Liang, J. Min, Q. Wu, S. Cheng, J. Ruan, M. Wang, Z. Shi, M. Wen, B. Liu, X. Ren, H. Zheng, D. Dong, K. Cook, G. Shan, H. Zhang, C. Kosiol, X. Xie, Z. Lu, H. Zheng, Y. Li, C. C. Steiner, T. T.-Y. Lam, S. Lin, Q. Zhang, G. Li, J. Tian, T. Gong, H. Liu, D. Zhang, L. Fang, C. Ye, J. Zhang, W. Hu, A. Xu, Y. Ren, G. Zhang, M. W. Bruford, Q. Li, L. Ma, Y. Guo, N. An, Y. Hu, Y. Zheng, Y. Shi, Z. Li, Q. Liu, Y. Chen, J. Zhao, N. Qu, S. Zhao, F. Tian, X. Wang, H. Wang, L. Xu, X. Liu, T. Vinar, Y. Wang, T.-W. Lam, S.-M. Yiu, S. Liu, H. Zhang, D. Li, Y. Huang, X. Wang, G. Yang, Z. Jiang, J. Wang, N. Qin, L. Li, J. Li, L. Bolund, K. Kristiansen, G. K.-S. Wong, M. Olson, X. Zhang, S. Li, H. Yang, J. Wang, and J. Wang, “The sequence and de novo assembly of the giant panda genome.,,” *Nature*, vol. 463, no. 7279, pp. 311–317, 2010.
- [52] R. Li, Y. Li, H. Zheng, R. Luo, H. Zhu, Q. Li, W. Qian, Y. Ren, G. Tian, J. Li, G. Zhou, X. Zhu, H. Wu, J. Qin, X. Jin, D. Li, H. Cao, X. Hu, H. Blanche, H. Cann, X. Zhang, S. Li, L. Bolund, K. Kristiansen, H. Yang, J. Wang, and J. Wang, “Building the sequence map of the human pan-genome.,,” *Nat. Biotechnol.*, vol. 28, no. 1, pp. 57–63, 2010.

- [53] J. C. Venter, M. D. Adams, E. W. Myers, P. W. Li, R. J. Mural, G. G. Sutton, H. O. Smith, M. Yandell, C. A. Evans, R. A. Holt, J. D. Gocayne, P. Amanatides, R. M. Ballew, D. H. Huson, J. R. Wortman, Q. Zhang, C. D. Kodira, X. H. Zheng, L. Chen, M. Skupski, G. Subramanian, P. D. Thomas, J. Zhang, G. L. Gabor Miklos, C. Nelson, S. Broder, A. G. Clark, J. Nadeau, V. A. McKusick, N. Zinder, A. J. Levine, R. J. Roberts, M. Simon, C. Slayman, M. Hunkapiller, R. Bolanos, A. Delcher, I. Dew, D. Fasulo, M. Flanigan, L. Florea, A. Halpern, S. Hannenhalli, S. Kravitz, S. Levy, C. Mobarry, K. Reinert, K. Remington, J. Abu-Threideh, E. Beasley, K. Biddick, V. Bonazzi, R. Brandon, M. Cargill, I. Chandramouliswaran, R. Charlab, K. Chaturvedi, Z. Deng, V. Di Francesco, P. Dunn, K. Eilbeck, C. Evangelista, A. E. Gabrielian, W. Gan, W. Ge, F. Gong, Z. Gu, P. Guan, T. J. Heiman, M. E. Higgins, R. R. Ji, Z. Ke, K. A. Ketchum, Z. Lai, Y. Lei, Z. Li, J. Li, Y. Liang, X. Lin, F. Lu, G. V Merkulov, N. Milshina, H. M. Moore, A. K. Naik, V. A. Narayan, B. Neelam, D. Nusskern, D. B. Rusch, S. Salzberg, W. Shao, B. Shue, J. Sun, Z. Wang, A. Wang, X. Wang, J. Wang, M. Wei, R. Wides, C. Xiao, C. Yan, A. Yao, J. Ye, M. Zhan, W. Zhang, H. Zhang, Q. Zhao, L. Zheng, F. Zhong, W. Zhong, S. Zhu, S. Zhao, D. Gilbert, S. Baumhueter, G. Spier, C. Carter, A. Cravchik, T. Woodage, F. Ali, H. An, A. Awe, D. Baldwin, H. Baden, M. Barnstead, I. Barrow, K. Beeson, D. Busam, A. Carver, A. Center, M. L. Cheng, L. Curry, S. Danaher, L. Davenport, R. Desilets, S. Dietz, K. Dodson, L. Doupe, S. Ferriera, N. Garg, A. Gluecksmann, B. Hart, J. Haynes, C. Haynes, C. Heiner, S. Hladun, D. Hostin, J. Houck, T. Howland, C. Ibegwam, J. Johnson, F. Kalush, L. Kline, S. Koduru, A. Love, F. Mann, D. May, S. McCawley, T. McIntosh, I. McMullen, M. Moy, L. Moy, B. Murphy, K. Nelson, C. Pfannkoch, E. Pratts, V. Puri, H. Qureshi, M. Reardon, R. Rodriguez, Y. H. Rogers, D. Romblad, B. Ruhfel, R. Scott, C. Sitter, M. Smallwood, E. Stewart, R. Strong, E. Suh, R. Thomas, N. N. Tint, S. Tse, C. Vech, G. Wang, J. Wetter, S. Williams, M. Williams, S. Windsor, E. Winn-Deen, K. Wolfe, J. Zaveri, K. Zaveri, J. F. Abril, R. Guigó, M. J. Campbell, K. V Sjolander, B. Karlak, A. Kejariwal, H. Mi, B. Lazareva, T. Hatton, A. Narechania, K. Diemer, A. Muruganujan, N. Guo, S. Sato, V. Bafna, S. Istrail, R. Lippert, R. Schwartz, B. Walenz, S. Yooseph, D. Allen, A. Basu, J. Baxendale, L. Blick, M. Caminha, J. Carnes-Stine, P. Caulk, Y. H. Chiang, M. Coyne, C. Dahlke, A. Mays, M. Dombroski, M. Donnelly, D. Ely, S. Esparham, C. Fosler, H. Gire, S. Glanowski, K. Glasser, A. Glodek, M. Gorokhov, K. Graham, B. Gropman, M. Harris, J. Heil, S. Henderson, J. Hoover, D. Jennings, C. Jordan, J. Jordan, J. Kasha, L. Kagan, C. Kraft, A. Levitsky, M. Lewis, X. Liu, J. Lopez, D. Ma, W. Majoros, J. McDaniel, S. Murphy, M. Newman, T. Nguyen, N. Nguyen, M. Nodell, S. Pan, J. Peck, M. Peterson, W. Rowe, R. Sanders, J. Scott, M. Simpson, T. Smith, A. Sprague, T. Stockwell, R. Turner, E. Venter, M. Wang, M. Wen, D. Wu, M. Wu, A. Xia, A. Zandieh, and X. Zhu, “The sequence of the human genome.,” *Science*, vol. 291, no. 5507, pp. 1304–1351, 2001.
- [54] B. Karlik, “Soft Computing Methods in Bioinformatics: A Comprehensive Review,” *Math. Comput. Appl.*, vol. 18, no. 3, pp. 176–197, 2013.
- [55] S. Mitra and T. Acharya, *Data Mining: Multimedia, Soft Computing, and Bioinformatics*, 1st ed. Wiley-Interscience, 2003, p. 424.
- [56] S. Mitra and Y. Hayashi, “Bioinformatics with soft computing,” *IEEE Trans. Syst. Man Cybern. Part C (Applications Rev.)*, vol. 36, no. 5, pp. 616–635, Sep. 2006.
- [57] Z. Ezziane, “Applications of artificial intelligence in bioinformatics: A review,” *Expert Systems with Applications*, vol. 30, no. 1, pp. 2–10, 2006.

- [58] Arabinda Panda and Satchidananda Dehuri, “A short and updated review of Soft computing tools in bioinformatics,” in *National Conference on Dynamics and Prospects of Data Mining (DPDM-2012)*, 2012, pp. 32–38.
- [59] P. Larrañaga, B. Calvo, R. Santana, C. Bielza, J. Galdiano, I. Inza, J. A. Lozano, R. Armañanzas, G. Santafé, A. Pérez, and V. Robles, “Machine learning in bioinformatics.,” *Brief. Bioinform.*, vol. 7, no. 1, pp. 86–112, 2006.
- [60] S. Aerts, P. Van Loo, Y. Moreau, and B. De Moor, “A genetic algorithm for the detection of new cis-regulatory modules in sets of coregulated genes.,” *Bioinformatics*, vol. 20, no. 12, pp. 1974–1976, 2004.
- [61] J. M. Bower and H. Bolouri, *Computational Modeling of Genetic and Biochemical Networks (Computational Molecular Biology)*. The MIT Press, 2004.
- [62] M. Krallinger, R. A. A. Erhardt, and A. Valencia, “Text-mining approaches in molecular biology and biomedicine,” *Drug Discovery Today*, vol. 10, no. 6, pp. 439–445, 2005.
- [63] J. T. L. Wang, M. J. Zaki, H. T. T. Toivonen, and D. Shasha, *Data Mining in Bioinformatics*. Springer-Verlag London, UK, 2005, p. 340.
- [64] M. Schlosshauer and M. Ohlsson, “A novel approach to local reliability of sequence alignments.,” *Bioinformatics*, vol. 18, no. 6, pp. 847–854, 2002.
- [65] T. S. Masato Wayama, Katsutoshi Takahashi, “An Approach to Amino Acid Sequence Alignment Using a Genetic Algorithm,” *Genome Informatics*, vol. 6, pp. 122–123, 1996.
- [66] C. Notredame and D. G. Higgins, “SAGA: sequence alignment by genetic algorithm.,” *Nucleic Acids Res.*, vol. 24, no. 8, pp. 1515–1524, 1996.
- [67] C. Zhang and A. K. Wong, “A genetic algorithm for multiple molecular sequence alignment,” *Comput Appl Biosci*, vol. 13, no. 6, pp. 565–581, 1997.
- [68] C. Shyu, L. Sheneman, and J. A. Foster, “Multiple Sequence Alignment with Evolutionary Computation,” *Genet. Program. Evolvable Mach.*, vol. 5, no. 2, pp. 121–144, 2004.
- [69] J.-T. Horng, L.-C. Wu, C.-M. Lin, and B.-H. Yang, “A genetic algorithm for multiple sequence alignment,” *Soft Comput.*, vol. 9, no. 6, pp. 407–420, Apr. 2004.
- [70] Z.-J. Lee, S.-F. Su, C.-C. Chuang, and K.-H. Liu, “Genetic algorithm with ant colony optimization (GA-ACO) for multiple sequence alignment,” *Appl. Soft Comput.*, vol. 8, no. 1, pp. 55–78, Jan. 2008.
- [71] T. Hou, J. Wang, L. Chen, and X. Xu, “Automated docking of peptides and proteins by using a genetic algorithm combined with a tabu search.,” *Protein Eng.*, vol. 12, no. 8, pp. 639–648, 1999.
- [72] T. Murata and H. Ishibuchi, “Positive and negative combination effects of crossover and mutation operators in sequencing problems,” *Proc. IEEE Int. Conf. Evol. Comput.*, 1996.
- [73] J. D. Szustakowski and Z. Weng, “Protein structure alignment using a genetic algorithm.,” *Proteins*, vol. 38, no. 4, pp. 428–440, 2000.

- [74] K. H. Tsutomu, T. Yokoyama, and T. Shimizu, “Multiple Sequence Alignment by Genetic Algorithm,” *Genome Informatics*, vol. 11, pp. 317–318, 2000.
- [75] L. A. Anbarasu, P. Narayanasamy, and V. Sundararajan, “Multiple Sequence Alignment by Parallel Genetic Algorithms,” *Curr. Sci.*, vol. 78, no. 7, pp. 858–863, Nov. 1998.
- [76] I. Yoshihara, K. Yamamori, and M. Yasunaga, “A parallel hybrid genetic algorithm for multiple protein sequence alignment,” in *Proceedings of the 2002 Congress on Evolutionary Computation. CEC'02 (Cat. No.02TH8600)*, 2002, vol. 1, pp. 309–314.
- [77] C. Gaspin and T. Schier, “Genetic Algorithms for Genetic Mapping,” *Lect. NOTES Comput. Sci.*, vol. 1363, pp. 145–155, 1998.
- [78] N. Qian and T. J. Sejnowski, “Predicting the secondary structure of globular proteins using neural network models.,” *J. Mol. Biol.*, vol. 202, no. 4, pp. 865–884, 1988.
- [79] W. Zhong, G. Altun, X. Tian, R. Harrison, P. C. Tai, and Y. Pan, “Parallel protein secondary structure prediction schemes using Pthread and OpenMP over hyper-threading technology,” *J. Supercomput.*, vol. 41, no. 1, pp. 1–16, Feb. 2007.
- [80] Y. Huang and Y. Li, “Prediction of protein subcellular locations using fuzzy k-NN method.,” *Bioinformatics*, vol. 20, no. 1, pp. 21–28, 2004.
- [81] V. Maniezzo and A. Carbonaro, “ANTS heuristic for the frequency assignment problem,” *Futur. Gener. Comput. Syst.*, vol. 16, no. 8, pp. 927–935, 2000.
- [82] K. J. Cios, H. Mamitsuka, T. Nagashima, R. Tadeusiewicz, O. Karpenko, J. Shi, and Y. Dai, “Prediction of MHC class II binders using the ant colony search strategy,” *Artif. Intell. Med.*, vol. 35, no. 1, pp. 147–156, 2005.
- [83] Y. Chen, Y. Pan, J. Chen, W. Liu, and L. Chen, *Multiple Sequence Alignment by Ant Colony Optimization and Divide-and-Conquer (Computational Science – ICCS 2006)*, vol. 3992. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 646–653.
- [84] W. Chen, B. Liao, W. Zhu, H. Liu, and Q. Zeng, “An ant colony pairwise alignment based on the dot plots.,” *J. Comput. Chem.*, vol. 30, no. 1, pp. 93–97, 2009.
- [85] A. Mikami and J. Shi, “A Modified Algorithm for Sequence Alignment Using Ant Colony System,” *Inf. Media Technol.*, vol. 4, no. 4, pp. 769–779, 2009.
- [86] S. I. Ao, K. Yip, M. Ng, D. Cheung, P.-Y. Fong, I. Melhado, and P. C. Sham, “CLUSTAG: hierarchical clustering and graph methods for selecting tag SNPs.,” *Bioinformatics*, vol. 21, no. 8, pp. 1735–1736, 2005.
- [87] K. Chellapilla and G. B. Fogel, “Multiple sequence alignment using evolutionary programming,” *Proc. 1999 Congr. Evol. Comput. Cat No 99TH8406*, vol. 1, no. 1 970, pp. 445–452, 1999.
- [88] et al. Ihsan Ömür Bucak, “Sequence alignment from the perspective of stochastic optimization: a survey,” *TURKISH J. Electr. Eng. Comput. Sci.*, vol. 19, no. 1, 2011.

- [89] P. F. Rodriguez, L. F. Niño, and O. M. Alonso, “Multiple sequence alignment using swarm intelligence,” *Int. J. Comput. Intell. Res.*, vol. 3, no. 2, pp. 123–129, 2007.
- [90] S.-F. S. Wang-Sheng Juang, “Multiple sequence alignment using modified dynamic programming and particle swarm optimization,” *J. Chinese Inst. Eng.*, vol. 31, pp. 659 – 673, 2008.
- [91] F. Xu and Y. Chen, “A method for multiple sequence alignment based on particle swarm optimization,” *Lect. Notes Comput. Sci.*, vol. 5755, pp. 965–973, Sep. 2009.
- [92] X.-J. Lei, J.-J. Sun, and Q.-Z. Ma, *Computational Intelligence and Intelligent Systems*, vol. 51. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, p. 351.
- [93] M. Ishikawa, T. Toya, M. Hoshida, K. Nitta, A. Ogiwara, and M. Kanehisa, “Multiple sequence alignment by parallel simulated annealing,” *Comput. Appl. Biosci.*, vol. 9, no. 3, pp. 267–273, 1993.
- [94] J. Kim, S. Pramanik, and M. J. Chung, “Multiple sequence alignment using simulated annealing.,” *Comput. Appl. Biosci.*, vol. 10, no. 4, pp. 419–426, 1994.
- [95] J. M. Keith, P. Adams, D. Bryant, D. P. Kroese, K. R. Mitchelson, D. A. E. Cochran, and G. H. Lala, “A simulated annealing algorithm for finding consensus sequences.,” *Bioinformatics*, vol. 18, no. 11, pp. 1494–1499, 2002.
- [96] M. Hernandez-Guia, R. Mulet, and S. Rodriguez-Perez, “Simulated annealing algorithm for the multiple sequence alignment problem: the approach of polymers in a random medium,” *Phys Rev E Stat Nonlin Soft Matter Phys*, vol. 72, no. 3 Pt 1, p. 31915, 2005.
- [97] W. Chen, B. Liao, W. Zhu, and X. Xiang, “Multiple sequence alignment algorithm based on a dispersion graph and ant colony algorithm.,” *J. Comput. Chem.*, vol. 30, no. 13, pp. 2031–2038, 2009.
- [98] A. Krogh, *Computational Methods in Molecular Biology*, vol. 32. Elsevier, 1998, pp. 45–63.
- [99] S. Tomida, T. Hanai, N. Koma, Y. Suzuki, T. Kobayashi, and H. Honda, “Artificial neural network predictive model for allergic disease using single nucleotide polymorphisms data,” *J. Biosci. Bioeng.*, vol. 93, no. 5, pp. 470–478, 2002.
- [100] M. D. Ritchie, B. C. White, J. S. Parker, L. W. Hahn, and J. H. Moore, “Optimization of neural network architecture using genetic programming improves detection and modeling of gene-gene interactions in studies of human diseases.,” *BMC Bioinformatics*, vol. 4, p. 28, 2003.
- [101] Y. Tomita, S. Tomida, Y. Hasegawa, Y. Suzuki, T. Shirakawa, T. Kobayashi, and H. Honda, “Artificial neural network approach for selection of susceptible single nucleotide polymorphisms and construction of prediction model on childhood allergic asthma.,” *BMC Bioinformatics*, vol. 5, p. 120, 2004.
- [102] E. Lin, Y. Hwang, S. C. Wang, Z. J. Gu, and E. Y. Chen, “An artificial neural network approach to the drug efficacy of interferon treatments,” *Pharmacogenomics.*, vol. 7, no. 7, pp. 1017–1024, 2006.

- [103] D. Curtis, “Comparison of artificial neural network analysis with other multimarker methods for detecting genetic association.,” *BMC Genet.*, vol. 8, p. 49, 2007.
- [104] Y. V Sun and S. L. Kardia, “Imputing missing genotypic data of single-nucleotide polymorphisms using neural networks.,” *Eur. J. Hum. Genet.*, vol. 16, no. 4, pp. 487–495, 2008.
- [105] P. Yang and Z. Zhang, *A Hybrid Approach to Selecting Susceptible Single Nucleotide Polymorphisms for Complex Disease Analysis*. IEEE, 2008, pp. 214–218.
- [106] H. Shi, Y. Lu, J. Du, W. Du, X. Ye, X. Yu, J. Ma, J. Cheng, Y. Gao, Y. Cao, L. Zhou, and Q. Li, “Application of back propagation artificial neural network on genetic variants in adiponectin ADIPOQ, peroxisome proliferator-activated receptor- γ , and retinoid X receptor- α genes and type 2 diabetes risk in a Chinese Han population.,” *Diabetes Technol. Ther.*, vol. 14, no. 3, pp. 293–300, Mar. 2012.
- [107] B. Karlik and E. Oztoprak, “ANN Based Application of Pharmacogenetics to Personalized Cancer Treatment,” *Int. Proc. Comput. Sci. Inf. Tech*, vol. 25, p. 132, 2012.
- [108] P. H. Lee and H. Shatkay, “BNTagger: improved tagging SNP selection using Bayesian networks.,” *Bioinformatics*, vol. 22, no. 14, pp. e211–e219, 2006.
- [109] B. Karlik and E. Öztoprak, “Personalized Cancer Treatment by Using Naive Bayes Classifier,” *Int. J. Mach. Learn. Comput.*, vol. 2, no. 3, pp. 339–344, 2012.
- [110] L. C.-L. Lin Min-Hui, “A Hybrid PSO-SVM Approach for Haplotype Tagging SNP Selection Problem,” *Int. J. Comput. Sci. Inf. Secur.*, 2010.
- [111] İ. İlhan, Y. Göktepe, and K. Ş., “Tag SNP selection using GA–SVM approach,” in *Proceedings of the IADIS European conference on data mining*, 2011, pp. 27–34.
- [112] G. Mahdevar, J. Zahiri, M. Sadeghi, A. Nowzari-Dalini, and H. Ahrabian, “Tag SNP selection via a genetic algorithm,” *J. Biomed. Inform.*, vol. 43, no. 5, pp. 800–804, 2010.
- [113] L. Chuang, Y. Hou, and C. Yang, “A Novel Prediction Method for Tag SNP Selection using Genetic Algorithm based on KNN,” *Eng. Technol.*, pp. 1325–1330, 2009.
- [114] I. İlhan and G. Tezel, “A genetic algorithm-support vector machine method with parameter optimization for selecting the tag SNPs,” *J. Biomed. Inform.*, vol. 46, no. 2, pp. 328–340, 2013.
- [115] C.-H. Y. C.-H. Yang, C.-H. H. C.-H. Ho, and L.-Y. C. L.-Y. Chuang, “Improved tag SNP selection using binary particle swarm optimization,” *2008 IEEE Congr. Evol. Comput. (IEEE World Congr. Comput. Intell.)*, 2008.
- [116] C. Yang, Z. He, X. Wan, Q. Yang, H. Xue, and W. Yu, “SNPHarvester: a filtering-based approach for detecting epistatic interactions in genome-wide association studies.,” *Bioinformatics*, vol. 25, no. 4, pp. 504–511, 2009.
- [117] M. H. Aghdam, N. Ghasem-Aghaee, and M. Ehsan Basiri, “Application of ant colony optimization for feature selection in text categorization,” in *2008 IEEE Congress on*

Evolutionary Computation (IEEE World Congress on Computational Intelligence), 2008, pp. 2867–2873.

- [118] Z. Lin and R. B. Altman, “Finding haplotype tagging SNPs by use of principal components analysis.,” *Am. J. Hum. Genet.*, vol. 75, no. 5, pp. 850–861, 2004.
- [119] S. Petrovski, C. E. Szoek, L. J. Sheffield, W. D’souza, R. M. Huggins, and T. J. O’brien, “Multi-SNP pharmacogenomic classifier is superior to single-SNP models for predicting drug outcome in complex diseases.,” *Pharmacogenet. Genomics*, vol. 19, no. 2, pp. 147–152, 2009.
- [120] L. I. Nahlawi and P. Mousavi, “Single nucleotide polymorphism selection using independent component analysis.,” *Conf. Proc. IEEE Eng. Med. Biol. Soc.*, vol. 2010, pp. 6186–6189, 2010.
- [121] V. Brusic, P. van Endert, J. Zelezniakow, S. Daniel, J. Hammer, and N. Petrovsky, “A neural network model approach to the study of human TAP transporter.,” *In Silico Biol.*, vol. 1, no. 2, pp. 109–121, 1999.
- [122] K. Chen and L. Kurgan, “PFRES: protein fold classification by using evolutionary information and predicted secondary structure.,” *Bioinformatics*, vol. 23, no. 21, pp. 2843–2850, 2007.
- [123] E. E. Snyder and G. D. Stormo, “Identification of protein coding regions in genomic DNA.,” *J. Mol. Biol.*, vol. 248, no. 1, pp. 1–18, 1995.
- [124] H. Saigo, J.-P. Vert, N. Ueda, and T. Akutsu, “Protein homology detection using string alignment kernels.,” *Bioinformatics*, vol. 20, no. 11, pp. 1682–1689, 2004.
- [125] J. Moult, K. Fidelis, A. Kryshtafovych, B. Rost, T. Hubbard, and A. Tramontano, “Critical assessment of methods of protein structure prediction-Round VII.,” *Proteins*, vol. 69 Suppl 8, pp. 3–9, 2007.
- [126] “The Universal Protein Resource (UniProt).,” *Nucleic Acids Res.*, vol. 35, no. Database issue, pp. D193–7, Jan. 2007.
- [127] C. H. Q. Ding and I. Dubchak, “Multi-class protein fold recognition using support vector machines and neural networks,” *Bioinformatics*, vol. 17, no. 4, pp. 349–358, 2001.
- [128] L. Tari, C. Baral, and S. Kim, “Fuzzy c-means clustering with prior biological knowledge,” *J. Biomed. Inform.*, vol. 42, no. 1, pp. 74–81, 2009.
- [129] R. Blankenbecler, M. Ohlsson, C. Peterson, and M. Ringner, “Matching protein structures with fuzzy alignments.,” *Proc. Natl. Acad. Sci. U. S. A.*, vol. 100, no. 21, pp. 11936–11940, 2003.
- [130] D. Wang, N. Lee, and T. S. Dillon, “Extraction and Optimization of Fuzzy Protein Sequences Classification Rules Using GRBF Neural Networks,” *Neural Inf. Process. - Lett. Rev.*, vol. 1, no. 1, pp. 53–59, 2003.
- [131] L. Zadeh, “Fuzzy logic, neural networks, and soft computing,” *Commun. ACM*, vol. 37, no. 3, pp. 77–84, 1994.

- [132] N. Krasnogor, W. E. Hart, J. Smith, and D. A. Pelta, “Protein structure prediction with evolutionary algorithms,” in *Proceedings of the Genetic and Evolutionary Computation Conference*, 1999, vol. 2, pp. 1596–1601.
- [133] A. Dal Palù, A. Dovier, and F. Fogolari, “Constraint Logic Programming approach to protein structure prediction.,” *BMC Bioinformatics*, vol. 5, p. 186, 2004.
- [134] L. Liu, Y. Fang, M. Li, and C. Wang, “Prediction of beta-turn in protein using E-SSpred and support vector machine.,” *Protein J.*, vol. 28, no. 3–4, pp. 175–181, 2009.
- [135] G. Pollastri, A. J. M. Martin, C. Mooney, and A. Vullo, “Accurate prediction of protein secondary structure and solvent accessibility by consensus combiners of sequence and structure information.,” *BMC Bioinformatics*, vol. 8, p. 201, 2007.
- [136] Z. Xiu-fen, P. Zi-shu, K. Li-shan, and Z. Chu-yu, “The evolutionary computation techniques for protein structure prediction: A survey,” *Wuhan Univ. J. Nat. Sci.*, vol. 8, no. 1, pp. 297–302, Mar. 2003.
- [137] A. P. Engelbrecht, *Fundamentals of Computational Swarm Intelligence*, vol. 8, no. 1. Wiley, 2005, pp. 3–4.
- [138] N. McCarthy, “Tumour profiling: Networking, protein style,” *Nat. Rev. Cancer*, vol. 7, no. 12, pp. 892–893, Dec. 2007.
- [139] Y. Zhang and J. S. Liu, “Bayesian inference of epistatic interactions in case-control studies.,” *Nat. Genet.*, vol. 39, no. 9, pp. 1167–1173, 2007.
- [140] M. C. Hinestrosa, K. Dickersin, P. Klein, M. Mayer, K. Noss, D. Slamon, G. Sledge, and F. M. Visco, “Shaping the future of biomarker research in breast cancer to ensure clinical relevance.,” *Nat. Rev. Cancer*, vol. 7, no. 4, pp. 309–315, 2007.
- [141] A. G. Hatzigeorgiou and M. Reckzo, “Signal peptide prediction on DNA sequences with artificial neural networks,” *IEEE Int. Work. Biomed. Circuits Syst. 2004.*, 2004.
- [142] V. Brusic, G. Rudy, G. Honeyman, J. Hammer, and L. Harrison, “Prediction of MHC class II-binding peptides using an evolutionary algorithm and artificial neural network.,” *Bioinformatics*, vol. 14, no. 2, pp. 121–130, 1998.
- [143] A. Narayanan, E. C. Keedwell, and B. Olsson, “Artificial intelligence techniques for bioinformatics.,” *Appl. Bioinformatics*, vol. 1, no. 4, pp. 191–222, 2002.
- [144] A. Torres and J. J. Nieto, “The fuzzy polynucleotide space: basic properties.,” *Bioinformatics*, vol. 19, no. 5, pp. 587–592, 2003.
- [145] M. Xiong, J. Li, and X. Fang, “Identification of genetic networks.,” *Genetics*, vol. 166, no. 2, pp. 1037–1052, 2004.
- [146] J. Cheng and P. Baldi, “Improved residue contact prediction using support vector machines and a large feature set.,” *BMC Bioinformatics*, vol. 8, p. 113, 2007.

- [147] J. W. Fickett and M. Cinkosky, “A genetic algorithm for assembling chromosome physical maps,” in *Second International Conference on Bioinformatics, Supercomputing, and Complex Genome Analysis*, 1993, pp. 272–285.
- [148] H. Murao, H. Tamaki, and S. Kitamura, “A coevolutionary approach to adapt the genotype-phenotype map in genetic algorithms,” *Proc. 2002 Congr. Evol. Comput. CEC'02 (Cat. No.02TH8600)*, vol. 2, 2002.
- [149] J. W. Fickett, “Finding genes by computer: The state of the art,” *Trends in Genetics*, vol. 12, no. 8, pp. 316–320, 1996.
- [150] A. Kel, A. Ptitsyn, V. Babenko, S. Meier-Ewert, and H. Lehrach, “A genetic algorithm for designing gene family-specific oligonucleotide sets used for hybridization: the G protein-coupled receptor protein superfamily.,” *Bioinformatics*, vol. 14, no. 3, pp. 259–270, 1998.
- [151] V. G. Levitsky and A. V. Katokhin, “Recognition of eukaryotic promoters using a genetic algorithm based on iterative discriminant analysis,” *Silico Biol*, vol. 3, pp. 81–7, 2003.
- [152] S. Knudsen, “Promoter2.0: for the recognition of PolII promoter sequences.,” *Bioinformatics*, vol. 15, no. 5, pp. 356–361, 1999.
- [153] N. M. Luscombe, D. Greenbaum, and M. Gerstein, “What is bioinformatics? A proposed definition and overview of the field.,” *Methods Inf. Med.*, vol. 40, no. 4, pp. 346–358, 2001.
- [154] G. B. Fogel and D. W. Corne, *Evolutionary Computation in Bioinformatics*, 1st ed. Morgan Kaufmann Publishers Inc., 2002, p. 393.
- [155] B. Qian, S. Raman, R. Das, P. Bradley, A. J. McCoy, R. J. Read, and D. Baker, “High-resolution structure prediction and the crystallographic phase problem.,” *Nature*, vol. 450, no. 7167, pp. 259–264, 2007.
- [156] D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, vol. Addison-We. Addison-Wesley, 1989, p. 432.
- [157] C. Carleos, F. Rodriguez, H. Lamelas, and J. A. Baro, “Simulating complex traits influenced by genes with fuzzy-valued effects in pedigreed populations.,” *Bioinformatics*, vol. 19, no. 1, pp. 144–148, 2003.
- [158] D. Dembélé and P. Kastner, “Fuzzy C-means method for clustering microarray data.,” *Bioinformatics*, vol. 19, no. 8, pp. 973–980, 2003.
- [159] A. Heger and L. Holm, “Sensitive pattern discovery with ‘fuzzy’ alignments of distantly related proteins.,” *Bioinformatics*, vol. 19 Suppl 1, pp. i130–i137, 2003.
- [160] P. J. Woolf and Y. Wang, “A fuzzy logic approach to analyzing gene expression data.,” *Physiol. Genomics*, vol. 3, no. 1, pp. 9–15, 2000.
- [161] S. Bicciato, M. Pandin, and C. Di Bello, “Analysis Of An Associative Memory Neural Network For Pattern Identification In Gene Expression Data,” *BIOKDD*, pp. 22–30, 2001.
- [162] S. Tomida, T. Hanai, H. Honda, and T. Kobayashi, “Analysis of expression profile using fuzzy adaptive resonance theory,” *Bioinformatics*, vol. 18, no. 8, pp. 1073–1083, 2002.

- [163] C. Branden and J. Tooze, *Introduction to Protein Structure*, 2nd ed. Garland Science, 1999, p. 410.
- [164] H.-K. Tsai, J.-M. Yang, and C.-Y. Kao, “Applying Genetic Algorithms to Finding the Optimal Gene Order in Displaying the Microarray Data,” in *GECCO '02 Proceedings of the Genetic and Evolutionary Computation Conference*, 2002, pp. 610–617.
- [165] J. Quackenbush, “Computational analysis of microarray data.,” *Nat. Rev. Genet.*, vol. 2, no. 6, pp. 418–427, 2001.
- [166] H.-K. Tsai, J.-M. Yang, Y.-F. Tsai, and C.-Y. Kao, “An evolutionary approach for gene expression patterns.,” *IEEE Trans. Inf. Technol. Biomed.*, vol. 8, no. 2, pp. 69–78, Jun. 2004.
- [167] A. S. Wu and I. Garibay, “The Proportional Genetic Algorithm: Gene Expression in a Genetic Algorithm,” *Genet. Program. Evolvable Mach.*, vol. 3, no. 2, pp. 157–192, 2002.
- [168] T. Akutsu, S. Miyano, and S. Kuhara, “Identification of genetic networks from a small number of gene expression patterns under the Boolean network model.,” *Pac. Symp. Biocomput.*, pp. 17–28, 1999.
- [169] Т. Атанасова, *Интелигентни компютърни системи*, Второ изда. Варна: Наука и икономика, 2011.
- [170] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, “Learning representations by back-propagating errors,” *Nature*, vol. 323, no. 6088. pp. 533–536, 1986.
- [171] M. J. Atallah, *Algorithms and Theory of Computation Handbook*. CRC Press, 1998, p. 1312.
- [172] R. Rojas, *Neural networks: a systematic introduction*. Springer, 1996, p. 502.
- [173] M. J. Atallah and M. Blanton, *Algorithms and Theory of Computation Handbook*, 2nd ed., vol. 1. Chapman and Hall/CRC, 2009, p. 988.
- [174] J. V. Ramos, C. Goncalves, and A. Dourado, *Artificial Intelligence Applications and Innovations*, vol. 154. Boston: Kluwer Academic Publishers, 2004, p. 87.
- [175] M. Mohri, A. Rostamizadeh, and A. Talwalkar, *Foundations of Machine Learning*. The MIT Press, 2012, p. 432.
- [176] J. Vermorel and M. Mohri, *Multi-armed bandit algorithms and empirical evaluation*. Springer, 2005, pp. 437 – 448.
- [177] B. Fritzke, “Some Competitive Learning Methods.”
- [178] C. B. Lucasius and G. Kateman, “Application of genetic algorithms in chemometrics,” in *Proceedings of the third international conference on Genetic algorithms*, 1989, pp. 170–176.
- [179] R. J. Parsons, S. Forrest, and C. Burks, “Genetic Algorithms for DNA Sequence Assembly,” in *Proceedings of the 1st International Conference on Intelligent Systems for Molecular Biology*, 1993, pp. 310–318.

- [180] R. J. Parsons, S. Forrest, and C. Burks, “Genetic algorithms, operators, and DNA fragment assembly,” *Mach. Learn.*, vol. 21, no. 1–2, pp. 11–33, Oct. 1995.
- [181] M. Mitchell, *An Introduction to Genetic Algorithms*. MIT Press, 1995, pp. 1–17.
- [182] S. Bandyopadhyay, *Analysis of Biological Data: A Soft Computing Approach*, vol. 3. World Scientific Publishing Co., Inc., 2007.
- [183] D. E. Goldberg and J. Richardson, “Genetic algorithms with sharing for multimodal function optimization,” in *Proceedings of the Second International Conference on Genetic Algorithms*, 1987, vol. 1, no. 3, pp. 41–49.
- [184] D. Heckerman, “Bayesian Networks for Data Mining,” *Data Min. Knowl. Discov.*, vol. 1, no. 1, pp. 79–119, Jan. 1997.
- [185] Wikipedia, “Variable elimination.” [Online]. Available: http://en.wikipedia.org/wiki/Variable_elimination. [Accessed: 02-Apr-2014].
- [186] F. Jensen, S. Lauritzen, and K. Olesen, “Bayesian updating in causal probabilistic networks by local computations,” *Comput. Stat. Quaterly*, vol. 4, pp. 269 – 282, 1990.
- [187] S. L. Lauritzen and D. J. Spiegelhalter, “Local computations with probabilities on graphical structures and their application to expert systems,” pp. 415–448, Jun. 1990.
- [188] M. Pradhan, G. Provan, B. Middleton, and M. Henrion, “Knowledge engineering for large belief networks,” pp. 484–490, Jul. 1994.
- [189] A. Darwiche, “Recursive conditioning,” *Artif. Intell.*, vol. 126, no. 1–2, pp. 5–41, 2001.
- [190] A. P. Dempster, N. M. Laird, and D. B. Rubin, “Maximum likelihood from incomplete data via the EM algorithm,” *J. R. Stat. Soc. Ser. B*, vol. 39, no. 1, pp. 1–38, 1977.
- [191] G. Rebane and J. Pearl, “The recovery of causal poly-trees from statistical data,” *Int. J. Approx. Reason.*, vol. 2, no. 3, pp. 341–342, 1988.
- [192] G. Shafer, *A Mathematical Theory of Evidence*. Princeton University Press, 1976, p. 314.
- [193] A. P. Dempster, “Upper and Lower Probabilities Induced by a Multivalued Mapping,” *The Annals of Mathematical Statistics*, vol. 38, no. 2. pp. 325–339, 1967.
- [194] T. L. Fine, “Review: Glenn Shafer, A mathematical theory of evidence,” *Bull. Am. Math. Soc.*, vol. 83, no. 4, pp. 667–672, Jul. 1977.
- [195] “Roche - Doing now what patients need next.” [Online]. Available: <http://www.roche.com/index.htm>. [Accessed: 02-Apr-2014].
- [196] M. Margulies, “Genome sequencing in microfabricated high-density picolitre reactors,” *Nature*, vol. 437, pp. 376–380, 2005.
- [197] Roche Company, “Products - GS FLX+ System□: 454 Life Sciences, a Roche Company.” [Online]. Available: <http://my454.com/products/gs-flx-system/index.asp>. [Accessed: 14-Feb-2014].

- [198] P. Bajgain, B. Richardson, J. Price, R. Cronn, and J. Udall, “Transcriptome characterization and polymorphism detection between subspecies of big sagebrush (*Artemisia tridentata*).,” *BMC Genomics*, 2011.
- [199] J. P. Délano-Frier, H. Avilés-Arnaut, K. Casarrubias-Castillo, G. Casique-Arroyo, P. A. Castrillón-Arbeláez, L. Herrera-Estrella, J. Massange-Sánchez, N. A. Martínez-Gallardo, F. I. Parra-Cota, E. Vargas-Ortiz, and M. G. Estrada-Hernández, “Transcriptomic analysis of grain amaranth (*Amaranthus hypochondriacus*) using 454 pyrosequencing: comparison with *A. tuberculatus*, expression profiling in stems and in response to biotic and abiotic stress.,” *BMC Genomics*, vol. 12, p. 363, 2011.
- [200] M. a Troncoso-Ponce, A. Kilaru, X. Cao, T. P. Durrett, J. Fan, J. K. Jensen, N. a Thrower, M. Pauly, C. Wilkerson, and J. B. Ohlrogge, “Comparative deep transcriptional profiling of four developing oilseeds.,” *Plant J.*, vol. 68, no. 6, pp. 1014–27, 2011.
- [201] A. P. M. Weber, K. L. Weber, K. Carr, C. Wilkerson, and J. B. Ohlrogge, “Sampling the *Arabidopsis* transcriptome with massively parallel pyrosequencing.,” *Plant Physiol.*, vol. 144, no. 1, pp. 32–42, 2007.
- [202] J. P. Hamilton and C. R. Buell, “Advances in plant genome sequencing.,” *Plant J.*, vol. 70, no. 1, pp. 177–90, 2012.
- [203] Roche Company, “Applications - Whole Genome Sequencing□: 454 Life Sciences, a Roche Company.” [Online]. Available: <http://454.com/applications/whole-genome-sequencing/index.asp>. [Accessed: 14-Feb-2014].
- [204] S. A. Filichkin, H. D. Priest, S. A. Givan, R. Shen, D. W. Bryant, S. E. Fox, W.-K. Wong, and T. C. Mockler, “Genome-wide mapping of alternative splicing in *Arabidopsis thaliana*.,” *Genome Res.*, vol. 20, no. 1, pp. 45–58, 2010.
- [205] G. Zhang, G. Guo, X. Hu, Y. Zhang, Q. Li, R. Li, R. Zhuang, Z. Lu, Z. He, X. Fang, L. Chen, W. Tian, Y. Tao, K. Kristiansen, X. Zhang, S. Li, H. Yang, J. Wang, and J. Wang, “Deep RNA sequencing at single base-pair resolution reveals high complexity of the rice transcriptome.,” *Genome Res.*, vol. 20, no. 5, pp. 646–654, 2010.
- [206] R. M. Davidson, C. N. Hansey, M. Gowda, K. L. Childs, H. Lin, B. Vaillancourt, R. S. Sekhon, N. de Leon, S. M. Kaepller, N. Jiang, and C. R. Buell, “Utility of RNA Sequencing for Analysis of Maize Reproductive Transcriptomes,” *The Plant Genome Journal*, vol. 4, no. 3. p. 191, 2011.
- [207] S. Huang, R. Li, Z. Zhang, L. Li, X. Gu, W. Fan, W. J. Lucas, X. Wang, B. Xie, P. Ni, Y. Ren, H. Zhu, J. Li, K. Lin, W. Jin, Z. Fei, G. Li, J. Staub, A. Kilian, E. A. G. van der Vossen, Y. Wu, J. Guo, J. He, Z. Jia, Y. Ren, G. Tian, Y. Lu, J. Ruan, W. Qian, M. Wang, Q. Huang, B. Li, Z. Xuan, J. Cao, Asan, Z. Wu, J. Zhang, Q. Cai, Y. Bai, B. Zhao, Y. Han, Y. Li, X. Li, S. Wang, Q. Shi, S. Liu, W. K. Cho, J.-Y. Kim, Y. Xu, K. Heller-Usszynska, H. Miao, Z. Cheng, S. Zhang, J. Wu, Y. Yang, H. Kang, M. Li, H. Liang, X. Ren, Z. Shi, M. Wen, M. Jian, H. Yang, G. Zhang, Z. Yang, R. Chen, S. Liu, J. Li, L. Ma, H. Liu, Y. Zhou, J. Zhao, X. Fang, G. Li, L. Fang, Y. Li, D. Liu, H. Zheng, Y. Zhang, N. Qin, Z. Li, G. Yang, S. Yang, L. Bolund, K. Kristiansen, H. Zheng, S. Li, X. Zhang, H. Yang, J. Wang, R. Sun, B. Zhang, S. Jiang, J. Wang, Y. Du, and S. Li, “The genome of the cucumber, *Cucumis sativus L.*,” *Nat. Genet.*, vol. 41, no. 12, pp. 1275–1281, 2009.

- [208] M. Dassanayake, D.-H. Oh, J. S. Haas, A. Hernandez, H. Hong, S. Ali, D.-J. Yun, R. A. Bressan, J.-K. Zhu, H. J. Bohnert, and J. M. Cheeseman, “The genome of the extremophile crucifer *Thellungiella parvula.*,” *Nat. Genet.*, vol. 43, no. 9, pp. 913–918, 2011.
- [209] T. P. G. Sequencing Consortium, “Genome sequence and analysis of the tuber crop potato.,” *Nature*, vol. 475, no. 7355, pp. 189–195, 2011.
- [210] X. Huang, Q. Feng, Q. Qian, Q. Zhao, L. Wang, A. Wang, J. Guan, D. Fan, Q. Weng, T. Huang, G. Dong, T. Sang, and B. Han, “High-throughput genotyping by whole-genome resequencing.,” *Genome Res.*, vol. 19, no. 6, pp. 1068–1076, 2009.
- [211] J. Cao, K. Schneeberger, S. Ossowski, T. Günther, S. Bender, J. Fitz, D. Koenig, C. Lanz, O. Stegle, C. Lippert, X. Wang, F. Ott, J. Müller, C. Alonso-Blanco, K. Borgwardt, K. J. Schmid, and D. Weigel, “Whole-genome sequencing of multiple *Arabidopsis thaliana* populations,” *Nature Genetics*, vol. 43, no. 10, pp. 956–963, 2011.
- [212] J. Lai, R. Li, X. Xu, W. Jin, M. Xu, H. Zhao, Z. Xiang, W. Song, K. Ying, M. Zhang, Y. Jiao, P. Ni, J. Zhang, D. Li, X. Guo, K. Ye, M. Jian, B. Wang, H. Zheng, H. Liang, X. Zhang, S. Wang, S. Chen, J. Li, Y. Fu, N. M. Springer, H. Yang, J. Wang, J. Dai, P. S. Schnable, and J. Wang, “Genome-wide patterns of genetic variation among elite maize inbred lines.,” *Nat. Genet.*, vol. 42, no. 11, pp. 1027–1030, 2010.

СПИСЪК НА ФИГУРИТЕ

Фигура 1: Файлови формати за данни от ДНК секвенции.....	13
Фигура 2: Видове матрици на субституция.....	20
Фигура 3: Генериране и Анализ на съпоставени секвенции	24
Фигура 4: J.R. Miller et al. / Genomics 95 (2010) 315–327 317	31
Фигура 5: 318 J.R. Miller et al. / Genomics 95 (2010) 315–327	32
Фигура 6: Сложност на K-тег-ните графи	33
Фигура 7: Видове функции	54
Фигура 8: Общ вид на изкуствения неврон	55
Фигура 9: Обща схема на ГА	59
Фигура 10: Подходи за отчитане на непълнотата на информацията	63
Фигура 11: Области на приложение на данните от паралелно секвениране.....	81
Фигура 12: Платформи за секвениране, приложение, качество	85
Фигура 13: Процеси при обработката на данни в авторската разработка	90
Фигура 14: Общ алгоритъм на асемблиране на къси прочити с МИРПИ	91
Фигура 15: Диаграма на преходите между различните библиотеки,.....	93
Фигура 16: Ирис 1	94
Фигура 17: Ирис 2	95
Фигура 19: Динуклеотиден профил с/д статистиките: "средно" и "стандартно отклонение".....	108
Фигура 20: Динуклеотиден профил с/д статистиките: "стандартно отклонение" и "SNR"....	108
Фигура 21: Динуклеотиден профил с/д статистиките: "средно" и "SNR"	108
Фигура 22: Динуклеотиден профил с/д статистиките: "стандартно отклонение" и "коefficient на вариация"	108
Фигура 23: Динуклеотиден профил с/д статистиките: "средно" и "коefficient на вариация"	108
Фигура 24: Динуклеотиден профил с/д статистиките: "SNR" и "коefficient на вариация" .	108
Фигура 25: Хомополимерен профил с/д статистиките "средно" и "стандартно отклонение"	108
Фигура 26: Хомополимерен профил с/д статистиките: "стандартно отклонени" и "SNR"	108
Фигура 27: Хомополимерен профил с/д статистиките: "средно" и "SNR"	108
Фигура 28: Хомополимерен профил с/д статистиките: "SNR" и "коefficient на вариация"	108
Фигура 29: Хомополимерен профил с/д статистиките: "средно" и "коefficient на вариация"	108
Фигура 30: Хомополимерен профил с/д статистиките: "SNR" и "коefficient на вариация"	108
Фигура 31: Профили между статистиките на Динуклеотидите или Хомополимерите (от една страна) и броя на N базите (от друга страна)	111
Фигура 32: Профилиране м/у Динуклеотидни х-ки и Хомополимерни х-ки.....	112

СПИСЪК НА ТАБЛИЦИТЕ

Таблица 1: Съотносимост между биоинформатичните задачи и методите на софт-компютинг, които се прилагат за тяхното решение.....	51
Таблица 2: Видове кръстосване	60
Таблица 3: Източници на непълнота и недостиг на данни	63
Таблица 4: Сравнителна таблица за технологиите на секвениране	82
Таблица 5: Секвенционни бази данни.....	85
Таблица 6: Софтуер за сравняване на секвенции	86
Таблица 7: Софтуер за асемблиране и поддържани технологии и типове на обработваните данни	87
Таблица 8: Основни характеристики на файловите формати.....	88
Таблица 9: Вид на трениращата таблица	96
Таблица 10: Таблица на Късите прочите и ИРПИ елементите.....	99
Таблица 11: Матрица за генериране на консенсусна верига	100
Таблица 12: Матрица за препокриващи се региони	101
Таблица 13: Информация, генерирана от overlap-ите	102
Таблица 14: Max K при различен процент на категория I в пътя	104
Таблица 15: Време за изчисление на 3-те профила на данните по брой N, *N, N*, N^k.....	107
Таблица 16: Имена на графиките за Динуклеотидното профилиране, в зависимост от подредбата им.....	108
Таблица 17: Имена на графиките за Хомополимерно профилиране, в зависимост от подредбата им.....	108
Таблица 18: Научни публикации.....	140
Таблица 19: Участия в научни форуми и проекти	142
Таблица 20: Участия в семинари и учебен процес по време на докторантурата	142
Таблица 21: Обучения по време на докторантурата	143