VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

ANALÝZA DAT Z DNA ČIPŮ

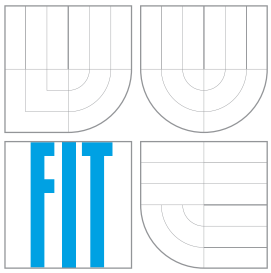
BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

IVAN VOGEL

BRNO 2008

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

ANALÝZA DAT Z DNA ČIPŮ

MICROARRAY DATA ANALYSIS

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — IVAN VOGEL
AUTHOR

VEDOUCÍ PRÁCE — Ing. JANA ŠILHAVÁ
SUPERVISOR

BRNO 2008

Abstrakt

Cieľom tejto bakalárskej práce je popísať technológiu a analýzu komplementárnych DNA čipov. Predstavené sú niektoré algoritmy využívané pri analýze. Podrobnejšie sa venuje problematike zhlukovania dát, predovšetkým rozoberá varianty hierarchického zhlukovania. Ďalej predstavuje prehľad dostupného komerčného aj nekomerčného softvéru z tejto oblasti. Zahŕňa návrh vlastnej implementácie hierarchických metód. V závere sú porovnané výsledky s podobným nekomerčným softvérom.

Klíčová slova

hybridizácia, DNA čip, zhlukovanie, dendrogram, matica podobností, korelačný koeficient

Abstract

The aim of this bachelor thesis is to describe the DNA microarray technology. Several analytical algorithms are discussed. It describes closely the clustering methods, especially several hierarchical clustering modifications. It gives a summary of available commercial and uncommercial software for microarray analysis. The thesis describes own implementation design of hierarchical methods. At the end, results are compared with similar uncommercial software.

Keywords

hybridisation, DNA microarray, clustering, dendrogram, similarity matrix, correlation coefficient

Citace

Ivan Vogel: Analýza dat z DNA čipů, bakalářská práce, Brno, FIT VUT v Brně, 2008

Analýza dat z DNA čipů

Prohlášení

Prehlasujem, že som túto bakalársku prácu vypracoval samostatne pod vedením Ing. Jany Šilhavej. Uviedol som všetky literárne pramene a publikácie, z ktorých som čerpal.

. . . . . .
Ivan Vogel
14. května 2008

Obsah

Kapitola 1

Úvod

Nedlhá existencia DNA čipov je dôkazom veľkého pokroku na poli génových technológií za posledné obdobie. Táto problematika vstupuje do popredia predovšetkým v súvislosti s rozmáhajúcimi sa civilizačnými ochoreniami. Skúmanie rozdielov medzi génovými expresiami chorých a zdravých buniek možno už čoskoro prinesie odpoveď na mnohé zatiaľ nezodpovedané otázky. DNA čip umožňuje na relatívne malej ploche umiestniť neuveriteľné množstvo génových expresií, čím ponúka vedcom úžasný nástroj. Technika je už známa dlhšie, no jednotlivé experimenty bolo doposiaľ vždy nutné robiť oddelene. Existuje veľké množstvo technologických postupov výroby, naviac sa tieto prudko rozvíjajú a modifikujú. Svoj pohľad som zameral na konkrétnu technológiu komplementárnych DNA čipov.

Samotný proces sa delí do niekoľkých fáz, ktoré budú v práci ďalej diskutované. Podrobnejšie rozvediem algoritmy zhlukovania dát získaných z DNA čipov, predovšetkým hierarchické zhlukovanie. Zdokumentujem moju implementáciu tejto metódy. Zmienim sa o dostupnom komerčnom aj nekomerčnom programovom vybavení, ktoré sa venuje tejto problematike. V závere sa pokúsim výsledky porovnať s výsledkami niektorého nekomerčného programu.

Kapitola 2

Spracovanie DNA čipov

V zásade existujú dva typy komerčne dostupných DNA čipov, a to tzv. sklené prefabrikátové DNA čipy, v ktorých sú zasadené jednotlivé fragmenty cDNA na sklenenú vrstvu. Druhým typom su tzv. oligonukleotidové biočipy s vysokou hustotou, ktoré spočívajú v syntéze oligonukleotidov in situ (napríklad technológia AffyMetrix GeneChips[13]). V rámci týchto dvoch veľkých skupín existuje celý rad rôznych technológií. Detailnejší popis týchto technologických postupov sa vymyká rozsahu práce, preto som sa zameral skôr na podrobnejší popis konkrétnej technológie, a to komplementárnych DNA čipov, ktoré patria do prvej uvedenej skupiny.

Táto kapitola sa zaoberá jednotlivými krokmi a postupmi výroby. Kľúčovými sú predovšetkým vo fáze fyzickej realizácie DNA čipu znalosti a technologické postupy z oblasti molekulárnej biológie a genetiky. Táto práca si určite nekladie za cieľ pokryť túto problematiku z biologického a biochemického hľadiska. Podáva však aspoň základné, častokrát čiastočne zjednodušené znalosti, ktorých štúdium bolo pre pochopenie celkovej technológie nevyhnutné.

2.1 Definícia základných termínov

Pre pochopenie technologických postupov je potrebný aspoň základný prehľad termínov z oblasti biológie a genetiky. Boli využité informácie z [2] a [5]:

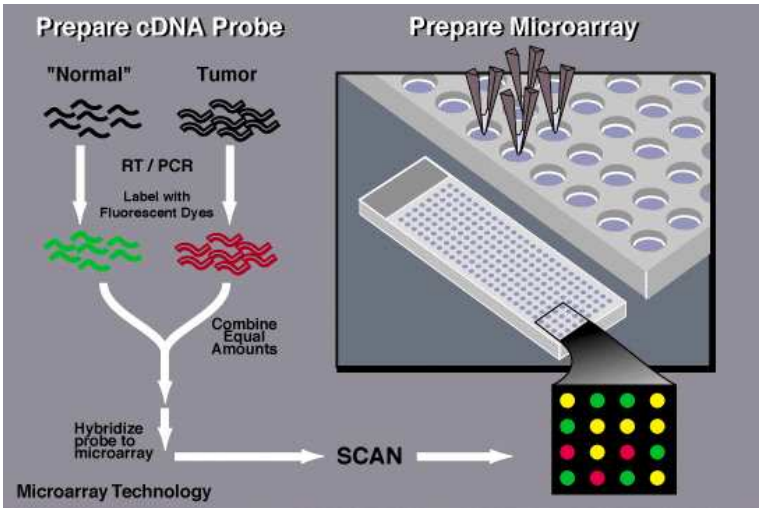
mRNA –vzniká transkripciou (prepisom) genetickej informácie podľa predlohy DNA. mRNA je biochemická štruktúra, podľa ktorej je následne syntetizovaná (v procese translácie) bielkovina ako stavebný materiál bunky. Celý tento proces je možno zjednodušene parafrázovať s prekladom programu na počítači. „Zdrojový kód“ v DNA je preložený do spustiteľnej podoby, ktorá generuje morfológiu a funkciu bunky. Súhrnne sa taktiež nazýva génová expresia (prejavenie sa génovej informácie v podobe nejakého konkrétneho produktu – RNA, resp. bielkoviny)

Komplementárna DNA –vzniká tzv. reverznou transkripciou mRNA do komplementárnej DNA (cDNA), čo je proces opačný k procesu transkripcie. Komplementárnu DNA si možno predstaviť ako jedno z vlákien DNA.

Hybridizácia –je proces párovania komplementárnych DNA zväzkov (cDNA). Práve vďaka tejto vlastnosti dusíkatých báz v DNA je možné zostrojiť DNA čip.

Bunkový fenotyp –je vzhľad bunky, to znamená vlastnosti genetickej informácie uloženej v bunke, ktoré sa prejavia ”navonok”.

2.2 Fáza výroby



Obrázek 2.1: Od výroby až po analýzu, obrázok prevzatý z [12]

2.2.1 Výber bunkovej populácie

Motiváciou pre vytvorenie experimentu s DNA čipom je porovnanie buniek, resp. génových expresií, ktoré sú v nich zakódované. Jeho konštrukcia umožňuje simultánne skúmať niekoľko stoviek takýchto génov. Ochorenia buniek, ktoré sú touto technikou skúmané, možno zaradiť medzi tzv. geneticky heterogénne. V praxi to znamená, že rovnaký defekt v bunkovom fenotype môže byť spôsobený dvoma úplne rozdielnymi génmi. Typickým príkladom takejto choroby je rakovina.

V zásade je potrebné pri experimente oddeliť tzv. referenčnú vzorku buniek (sú to bunky vyvinuté pri normálnych podmienkach, možno ich označiť za zdravé) a experimentálnu vzorku buniek, ktorá môže obsahovať skupiny rôznych buniek, akýmkoľvek spôsobom podozrivých z podielu na ochorení.

2.2.2 Extrakcia mRNA a reverzná transkripcia

K izolácii mRNA z bunky je potrebná dôkladná purifikácia bunkového obsahu. Problémom v tejto fáze je získanie dostatočného množstva mRNA, pretože tá tvorí častokrát len tri percentá celkového množstva RNA v bunke. Samotná mRNA je však pomerne nestabilná a náchylná k deštrukcii. Z tohto dôvodu sa pre účely experimentu prevedie na vzorke reverzná transkripcia, vďaka ktorej sa získa komplementárna DNA (cDNA). Tá je omnoho stabilnejšia.

2.2.3 Označkovanie cDNA

Aby bolo možné detekovať neskôr na čipe príslušnosť cDNA k referenčnej, resp. experimentálnej vzorke, je potrebné ju označiť identifikačnou molekulou. Je zaužívanou konvenciou, že pre referenčnú vzorku sa použije zelené farbivo (Cy3), zatiaľ čo pre experimentálnu vzorku červené farbivo (Cy5). Takto označkovanú cDNA nie je ešte v tejto fáze možné

akýmkoľvek spôsobom vizuálne rozlíšiť. Je potrebné ju najprv presvietiť laserom a následne odmerať odrazenú vlnovú dĺžku. Práve schopnosť odraziť svetlo určuje mieru fluorescencie príslušnej molekuly DNA.

2.2.4 Hybridizácia vzoriek do DNA čipu

Referenčná aj experimentálna vzorka sa následne hybridizujú do DNA čipu. Čip obsahuje stovky až tisíce bodov, kde každý obsahuje rozdielnu časť DNA sekvencie. Pokiaľ vzorka obsahuje cDNA, ktorá je komplementárna k sekvencii v danom bode DNA čipu, tak sa s týmto bodom hybridizuje. Vďaka fluorescenčnej molekule bude táto sekvencia v ďalších fázach detekovateľná. V tomto zmysle je každý bod na čipe nezávislým priestorom, ktorý je schopný poňať dostatočné množstvo cDNA z rôznych vzoriek bez akéhokoľvek vzájomného narušenia[2]. Hybridizáciou končí fyzická realizácia čipu a ďalej je vzniknutý produkt v niekoľkých fázach analyzovaný. Touto etapou sa zaoberá nasledujúca sekcia.

2.3 Fáza analýzy dát

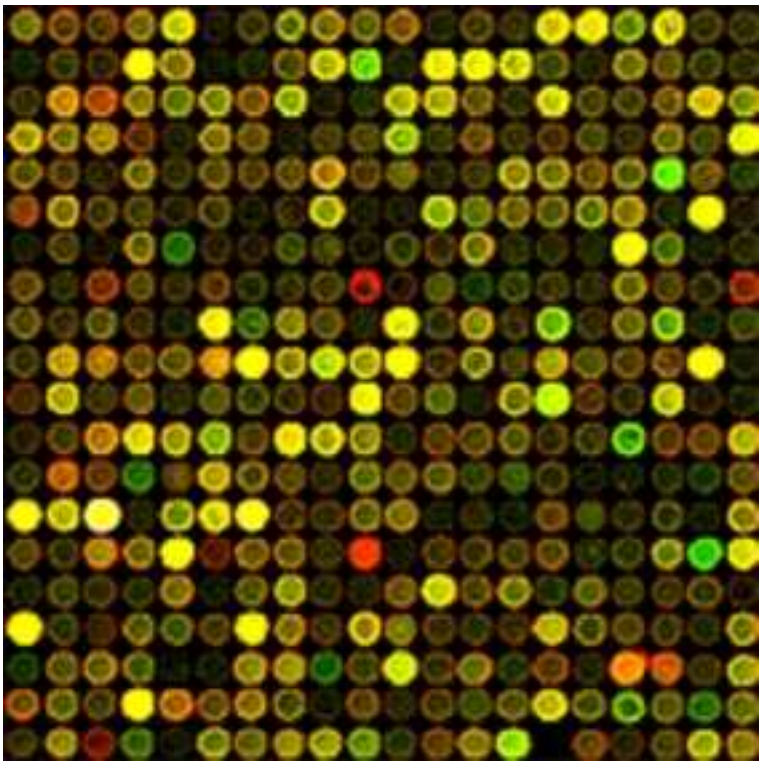
2.3.1 Skenovanie hybridizovaného čipu a jeho interpretácia

Po úspešnej hybridizácii je DNA čip podrobený skenovaniu. Už spomínané fluorescenčné molekuly naviazané na jednotlivé sekvencie cDNA sú špecifické svojou excitačnou vlnovou dĺžkou (t.j. vlnovou dĺžkou, pri ktorej sa „rozsvietia“).

Pri skenovaní je nutné použiť laser s dvomi rôznymi vlnovými dĺžkami korešpondujúcimi s excitačnou vlnovou dĺžkou fluorescenčných molekúl. Takýmto laserom osvetlený obraz sa zaznamená, presnejšie zaznamenajú sa dva obrazy – jeden pre "červené" molekuly, druhý pre "zelené". Intenzitu týchto svetelných bodov možno chápať ako kvantitatívnu analýzu jednotlivých cDNA tak, ako sa naviazali na jednotlivé sekvencie (body) DNA čipu. Čím viac cDNA molekúl sa naviazalo ku konkrétnemu bodu na čipe, tým bude mať tento bod pri laserovej emisii väčšiu intenzitu. Keď zvážime referenčnú vzorku s určitou intenzitou a experimentálnu vzorku s rovnakou hodnotou intenzity, potom ich zmiešaním dostávame žltú farbu. Pokiaľ intenzity nie sú rovnaké, farba má tendenciu približovať sa buď k zelenej alebo k červenej. Keď je intenzita červenej farby nižšia ako intenzita zelenej farby, hovoríme, že gén je potlačený (down-regulated), resp. nevýrazný, slabý. Znamená to, že sa v tomto experimente neprejavil, alebo sa prejavil len veľmi málo. V opačnom prípade hovoríme o nadmernom, silnom prejave génu (up-regulated). Situácia je zrejmá z obrázku 2.2, na ktorom sú obidva obrazy (obraz pre červené molekuly aj obraz pre zelené molekuly) spojené do jedného, aby bolo vidieť farebný efekt [2].

Z pomeru intenzít červenej a zelenej farby sa vypočíta tzv. relatívna úroveň génovej expresie v konkrétnom bode. Po prepočítaní všetkých hodnôt dostávame číselnú maticu génových expresií, kde riadok definuje konkrétny gén a stĺpec zasa konkrétny experiment.

Samotné nízkoúrovňové spracovanie obrazu pozostáva z troch základných fáz: mriežkovania (gridding), segmentácie (segmentation) a z extrakcie informácií (information extraction). Mriežkovanie spočíva v lokalizácii jednotlivých farebných bodov na čipe. Segmentácia diferencuje pixely v rámci konkrétnemu bodu na čipe a rozdeľuje ich na popredie (foreground, true signal) a pozadie (background). Extrakcia informácií má dve časti: extrakcia intenzity bodu a extrakcia intenzity okolia. Extrakcia intenzity okolia, resp. pozadia je dôležitá kvôli nešpecifickej hybridizácii, ktorá sa mohla na čipe uskutočniť a skreslila by



Obrázek 2.2: Obrázok naskenovaného hybridizovaného DNA čipu

výsledok. Najčastejšie je táto intenzita odčítaná od intenzity popredia, no existujú aj ďalšie metódy [3].

2.3.2 Predspracovanie dát

Predspracovanie dát zahŕňa niekoľko krokov.

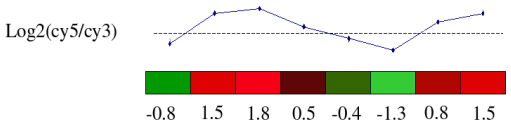
1. **Normalizácia**

   Po spracovaní obrazu DNA čipu máme k dispozícii červenú a zelenú fluorescenčnú intenzitu pre každý bod na DNA čipe. Pre zhlukovaciu analýzu je nutné sa uistiť, že rozdiely v intenzitách sú spôsobené rozdielnymi génovými expresiami, a nie nežiadanou hybridizáciou, chybnou interpretáciou obrazu atp. Existuje celý rad normalizačných metód, ktoré odstraňujú tieto systematické chyby a korigujú tým správnosť dát. Viac o týchto algoritmoch je možné nájsť napríklad v zdroji [15].

2. **Nelineárna transformácia**

   Nesymetrickosť získaných hodnôt je zrejmá zo spôsobu ich výpočtu pomocou pomeru intenzít referenčnej a experimentálnej vzorky. Pri génoch, ktorých intenzita je väčšia ako u referenčnej vzorky, sa hodnota pomeru pohybuje v rozmedzí jedna až nekonečno, pričom pri génoch s menšou intenzitou ako referenčnou sa hodnota pohybuje v rozmedzí nula až jedna. Logaritmovaním týchto hodnôt (zvyčajne sa používa dvojkový logaritmus) získame požadovanú symetrickosť vzhľadom k nule.

Po spracovaní a vyhodnotení obrazu DNA čipu má zmysel hovoriť o tzv. profile génovej expresie. Možno ho chápať ako vektor, ktorý zapúzdruje informácie o tom, do akej miery sa gén prejavil v rámci jednotlivých experimentov. Je to teda prechod všetkými hodnotami intenzít v rámci konkrétneho génu[9]. Graficky by sa dal vyjadriť nasledovne:



Obrázek 2.3: Profil génovej expresie

3. **Nahradenie chýbajúcej hodnoty a filtrácia**

   Z rôznych technických príčin môžu niektoré hodnoty chýbať. Mnoho algoritmov spracovania dát vyžaduje pre správny chod doplnenie týchto hodnôt. Jednoduché nahradenie číslom 0, prípadne priemerom hodnôt z ostatných experimentov častokrát skresľuje výsledok. Predovšetkým dosadenie priemeru niekedy vychádza z nesprávnej domnienky, že číselné hodnoty určitého génu sú v rámci všetkých experimentov podobné. Dokonalejšie techniky vychádzajú preto z pokročilejších metód[1]. Niektoré algoritmy spracovania DNA čipov chýbajúce hodnoty úplne vynechávajú a počítajú len s tými, ktoré existujú. Pri všetkej snahe o rekonštrukciu skreslených hodnôt je potrebné v závere predspracovania dát vyradiť hodnoty, ktorých validnosť zostáva otázna.

4. **Štandardizácia a zmena merítka (rescaling)**

   Biológovia sa zaujímajú predovšetkým o zoskupovanie profilov génových expresií s rovnakým tzv. relatívnym prejavom. To znamená, že je podstatná miera odklonu od referenčnej hodnoty a nie absolútna hodnota génovej expresie. Ide napríklad o také gény, ktoré majú rozdielnu amplitúdu profilu génovej expresie, ale ich profil génovej expresie prebieha podobne, t.j. génové expresie stúpajú a klesajú v rovnakých časoch. Pri zisťovaní podobnosti jednotlivých vektorov génových expresií metrikou Euklidovskej vzdialenosti by pri ich porovnávaní vyšla relatívne veľká vzdialenosť. Takémuto výsledku sa zamedzí štandardizáciou a zmenou merítka profilov génových expresií tak, aby mali nulový priemer a jednotkovú smerodatnú odchýlku[1].

Kapitola 3

Zhlukovacie algoritmy

3.1 Úvod

Pre ďalšie pozorovanie a skúmanie DNA čipu je potrebné dáta, ktorých získanie bolo predmetom predchádzajucej kapitoly, ďalej spracovať. Cieľom je zoskupiť jednotlivé gény na základe ich profilu génovej expresie do biologicky zmysluplných skupín. Na získanie týchto znalostí sa využívajú zhlukovacie algoritmy.

Všeobecne možno zhlukovacie metódy zaradiť medzi tzv. algoritmy strojového učenia. V závislosti od konkrétnej metódy rozlišujeme metódy bez učiteľa a s učiteľom [3]. V prípade metód bez učiteľa algoritmus nedostáva žiadnu informáciu o správnosti klasifikácie a ani o samotnom priebehu. Jedinou informáciou môže byť napríklad počet zhlukov, do ktorých má príklady z trénovacej množiny klasifikovať.

Naopak metódy s učiteľom sú charakteristické tým, že dostanú na začiatku určité množstvo informácií o dátach, ktoré môžu byť pri zhlukovaní využité. Algoritmy z tejto skupiny sú schopné učiť sa z informácií, ktoré definujú každý zhluk (väčšinou ide o trénovaciu množinu dát) a tieto vedomosti potom aplikovať na nové dáta pri zhlukovaní. Treba povedať, že zhlukovacie algoritmy bez učiteľa tvoria väčšiu skupinu, a preto budú v práci ďalej podrobnejšie diskutované.

Na základe spôsobu práce algoritmu a formy výstupu je možné vyčleniť dve skupiny zhlukovacích algoritmov, a to metódy hierarchické a nehierarchické. Hierarchické metódy sú založené na postupnom spájaní jednotlivých zhlukov do väčších celkov. Výsledkom týchto metód je binárny strom, tzv. dendrogram, z ktorého je zrejmá postupnosť jednotlivých zhlukovacích krokov.

Nehierarchické metódy zhlukovania sa na rozdiel od hierarchických pokúšajú vzory rozdeliť priamo, v jednom kroku, do niekoľkých zhlukov (pričom sa výsledok v každom kroku spresňuje). Ako výsledok teda nie je k dispozícii štruktúra zhlukov typu dendrogram. Nehierarchické metódy sa s výhodou používajú pri obrovskom množstve dát, kde by konštrukcia dendrogramu bola na výpočet príliš časovo a pamäťovo náročná.

V práci bude najväčšia časť venovaná klasickému hierarchickému zhlukovaniu, ktoré bolo ďalej predmetom mojej implementácie v rámci praktickej časti.

3.2 Metriky určovania podobnosti

3.2.1 Definície pojmov

Pri výklade princípov zhlukovacích algoritmov sú využívané pojmy prebraté z [6].

- Vzor (alebo vektor vlastností, pozorovanie) x – je dátový objekt využívaný zhlukovacím algoritmom. Typicky pozostáva z vektora d meraní: $x = (x_1, \cdots x_d)$
- Jednotlivé skalárne komponenty x_i vzoru x sa nazývajú vlastnosti alebo atribúty
- d je dimenzionalita vzoru
- množina vzorov sa značí $\kappa = x_1, \cdots x_n$. i-tý vzor sa značí ako $x_i = (x_{i,1}, \cdots x_{i,d})$ V mnohých prípadoch je za množinu vzorov, ktoré majú byť zhlukované, považovaná matica vzorov s rozmermi $n \times d$.

Zhlukovanie je proces, pri ktorom sa klasifikujú objekty, množiny dát do určitých podmnožín (zhlukov) na základe vzájomnej podobnosti. Informácia o podobnosti sa získava rôznymi metrikami odhadu vzájomnej vzdialenosti dvoch objektov. Výber konkrétnej metódy je dôležitý, závisí od povahy atribútov a od počtu dimenzií vzoru. K výpočtu sa využíva jav práve opačný k podobnosti, a to rozdielnosť vzorov (pattern dissimilarity). Tá je definovaná ich vzdialenosťou. V krátkosti predstavím vo forme matematických vzorcov niektoré významné metódy pre výpočet vzdialenosti dvoch vzorov resp. vektorov hodnôt [17].

Euklidovská vzdialenosť je definovaná ako:

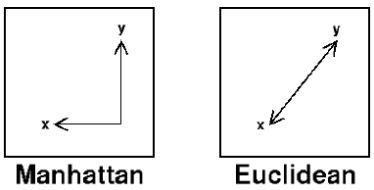
$$d_2(x_i, x_j) = \sqrt{\sum_{k=1}^{n} (x_{i,k} - x_{j,k})^2} \tag{3.1}$$

Je to najbežnejšia metrika zisťovania podobnosti. Ide vpodstate o klasický výpočet vzdialenosti dvoch bodov.

Manhattanovská vzdialenosť je definovaná ako:

$$d_2(x_i, x_j) = \sum_{k=1}^{n} |x_{i,k} - x_{j,k}| \tag{3.2}$$

Meria vzdialenosť dvoch bodov (objektov) ako dĺžku cesty, ktorá je vedená v horizontálnom plus vertikálnom smere. Z obrázku 3.1 je zrejmý rozdiel medzi euklidovskou a manhattanovskou vzdialenosťou.



Obrázek 3.1: Porovnanie princípu manhattanovskej a euklidovskej vzdialenosti

Chebyshevova vzdialenosť sa nazýva tiež maximálna hodnota vzdialenosti. Skúma absolútnu veľkosť rozdielov medzi atribútmi dvoch vzorov a vyberá najväčší z nich.

$$d_2(x_i, x_j) = max|x_{i,k} - x_{j,k}| \tag{3.3}$$

Minkowského vzdialenosť je generalizáciou predošlých metód určovania vzdialenosti [17]. Keď $\lambda = 1$ dostaneme Manhattanovskú vzdialenosť, keď $\lambda = 2$ získame Euklidovskú vzdialenosť. Chebyshevova vzdialenosť je špeciálnym prípadom, kde $\lambda = \infty$

$$d_2(x_i, x_j) = \sqrt[\lambda]{(\sum_{k=1}^{n} |x_{i,k} - x_{j,k}|^{\lambda})} \tag{3.4}$$

Pearsonov korelačný koeficient vyjadruje podobnosť dvoch vzorov odlišným spôsobom ako už spomínané metriky. Táto metóda určuje stupeň lineárneho vzťahu medzi dvoma premennými [14]. Vychádza zo vzorca:

$$r = \frac{1}{N} \sum_{i=1}^{N} \left(\frac{X_i - \bar{X}}{\sigma_X} \right) \left(\frac{Y_i - \bar{Y}}{\sigma_Y} \right) \tag{3.5}$$

σ_X resp. σ_Y udávajú smerodatnú odchýlku príslušného vzoru. Korelačný koeficient r sa pohybuje v rozmedzí od +1 do -1. Korelácia +1 znamená dokonale pozitívny lineárny vzťah. To v praxi znamená, že hodnoty z obidvoch porovnávaných skupín (v našom prípade dvoch vektorov génových expresií) zobrazené na grafe patria do množiny bodov jednej priamky, sú identické. Korelačný koeficient s hodnotou -1 potom znamená presný opak, teda dokonale negatívny lineárny vzťah. Objekty sú teda úplne odlišné. Podľa [4] uvedený vzorec 3.5 udáva výpočet tzv. centrovaného Pearsonovho korelačného koeficientu. Modifikáciou dostávame necentrovaný Pearsonov korelačný koeficient so vzorcom:

$$r = \frac{1}{N} \sum_{i=1}^{N} \left(\frac{X_i}{\sigma_X} \right) \left(\frac{Y_i}{\sigma_Y} \right) \tag{3.6}$$

Rozdiel medzi uvedenými funkciami spočíva v tom, že necentrovaná varianta uvažuje priemer hodnôt 0, aj v prípade, keď tomu tak nie je. Tento rozdiel sa prejaví v nasledovnej situácii. Predstavme si dva vektory X a Y, ktorých hodnoty na grafe dávajú identický tvar, ale sú od seba vzdialené určitou konštantnou vzdialenosťou. Takéto vektory by mali v prípade centrovaného Pearsonovho korelačného koeficientu hodnotu jedna, nie však v prípade necentrovaného korelačného koeficientu.

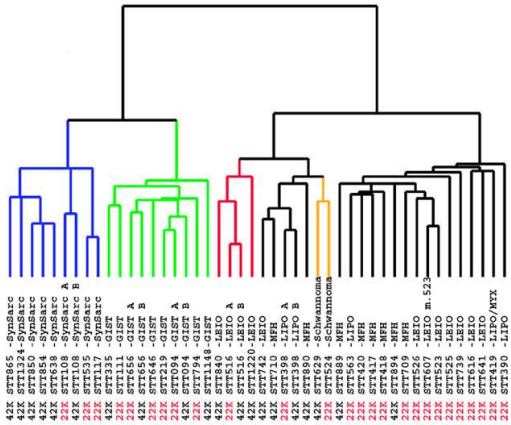
Výsledky uvedených metód je potrebné uložiť do vhodnej dátovej štruktúry. Väčšinou sa používa trojuholníková matica.

$$U = \begin{bmatrix} a_{00} & 0 & \cdots & 0 \\ a_{10} & a_{11} & \cdots & 0 \\ \vdots & \vdots & \ddots & 0 \\ a_{n0} & a_{n1} & \cdots & a_{nn} \end{bmatrix} \tag{3.7}$$

Platí, že vzdialenosť medzi i–tym a j–tym génom je uložená v prvku a_{ij}. Ďalej platí, že v prípade prvku a_{ij}, kde $i = j$, teda porovnania génu so sebou samým je vzdialenosť medzi nimi logicky nulová. Matica je zámerne indexovaná od nuly tak, aby korešpondovala s neskoršou návrhovou schémou v jazyku C++.

3.3 Hierarchické zhlukovanie

Hierarchické zhlukovanie je najrozšírenejšou metódou spracovania dát z DNA čipov. Jej nespornou výhodou je fakt, že výsledok algoritmu je možné názorne vizualizovať ako ukazuje obrázok 3.2. Negatívom je ich časová zložitosť, platí $O(n^2)$, kde n je počet vstupných objektov. Poznáme dva prístupy k tvorbe dendrogramu: prístup zhora – dole, tzv. divisive



Obrázek 3.2: Dendrogram

clustering a zdola – hore, tzv. aglomerative clustering. Práve druhým spomenutým sa budem zaoberať podrobnejšie.
Keď je na vstupe N objektov (vzorov) a matica podobností o rozmeroch $N \times N$, je základná štruktúra algoritmu klasického aglomeratívneho hierarchického zhlukovania nasledovná[10]:

1. Začni priradením každého vzoru do zhluku tak, že v prípade N vzorov dostaneš N zhlukov, pričom každý obsahuje práve jeden vzor. Nech vzdialenosti (alebo podobnosti) medzi vzniknutými zhlukmi sú rovnaké ako vzdialenosti medzi vzormi, ktoré obsahujú.
2. Nájdi najbližší (najpodobnejší) pár zhlukov a spoj ich do jedného zhluku tak, že máš celkovo o jeden zhluk menej.
3. Vypočítaj vzdialenosť medzi novým zhlukom a ostatnými zhlukmi.
4. Opakuj kroky 2 a 3 až kým nevznikne jediný zhluk s veľkosťou N.

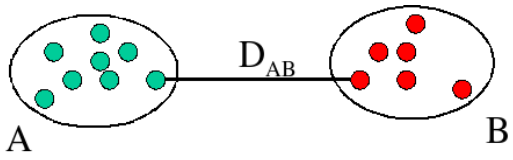
Krok 3 môže byť uskutočnený rôznymi spôsobmi, na základe čoho odlišujeme niekoľko variant algoritmu.

3.3.1 Metóda single linkage (najbližší sused)

Je jednou z najjednoduchších metód aglomeratívneho hierarchického zhlukovania. Vzdialenosť medzi dvoma zhlukmi je určená najkratšou možnou vzdialenosťou medzi ich členmi. Jej princíp znázorňuje obrázok 3.3. Platí:

$$D_{AB} = min(d(u_i, v_j)) \tag{3.8}$$

$$u \in A,\, v \in B;\, \forall i \in [1; N_A],\, \forall j \in [1; N_B] \tag{3.9}$$



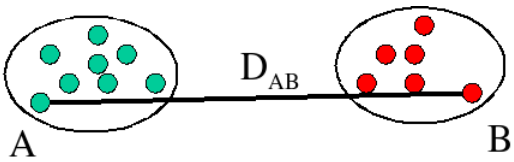
Obrázek 3.3: Princíp single linkage

3.3.2 Metóda complete linkage (najvzdialenejší sused)

Je opakom metódy single linkage, vzdialenosť dvoch porovnávaných zhlukov je teda určená najdlhšou vzdialenosťou medzi ich členmi. Situáciu znova znázorňuje obrázok 3.4. Platí:

$$D_{AB} = max(d(u_i, v_j)); \tag{3.10}$$

$$u \in A,\ v \in B;\ \forall i \in [1; N_A],\ \forall j \in [1; N_B] \tag{3.11}$$



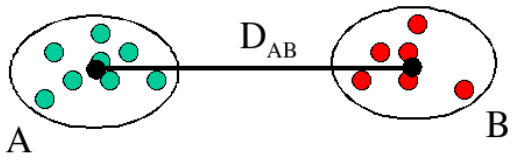
Obrázek 3.4: Princíp complete linkage

3.3.3 Metóda average linkage (priemerná vzdialenosť)

Vzdialenosť dvoch zhlukov sa počíta ako priemerná vzdialenosť všetkých vzájomných vzdialeností objektov, ktoré sú v nich obsiahnuté. K dispozícii je matematické vyjadrenie ako aj grafické znázornenie na obrázku 3.5.

$$D_{AB} = \frac{1}{N_A N_B} \sum_{i=1}^{N_A} \sum_{j=1}^{N_B} (d(u_i, v_j)); \tag{3.12}$$

$$u \in A,\ v \in B;\ \forall i \in [1; N_A] \wedge \forall j \in [1; N_B] \tag{3.13}$$



Obrázek 3.5: Princíp average linkage

3.3.4 Metóda average group linkage (priemerný vektor)

Popisuje vzdialenosť dvoch zhlukov ako vzdialenosť ich priemerných vektorov. Znamená to, že pre každý z dvoch zhlukov je potrebné najprv vypočítať priemerný vektor jeho hodnôt, tieto priemerné vektory následne porovnať a vyhodnotiť mieru podobnosti.

$$D_{AB} = d(\bar{u}, \bar{v}); \tag{3.14}$$

$$\bar{u} = \frac{1}{N_A} \sum_{i=1}^{N_A} u_i \tag{3.15}$$

$$\bar{v} = \frac{1}{N_B} \sum_{i=1}^{N_B} v_i \tag{3.16}$$

Vzhľadom k tomu, že posledné dve metódy dávajú zväčša totožné výsledky (v závislosti od typu metriky určovania podobnosti vzorov), budem ďalej v práci považovať metódu average group linkage za metódu average linkage. Tak isto sa k tejto problematike stavia aj autor v zdroji [4].

3.4 Nehierarchické zhlukovanie

V úvode tejto kapitoly už bol naznačený princíp nehierarchického zhlukovania. Snaží sa o dekompozíciu dátovej množiny do disjunktných zhlukov v jednom kroku. Nehierarchické zhlukovanie delíme do nasledovných kategórií [8] :

1. **Metódy Single-pass** vytvárajú zhluky, ktoré sú závislé od poradia vstupných objektov;
2. **Relocation metódy** ako napr. k-means. Cieľom je vytvárať optimálny rozklad objektov s vopred určeným počtom zhlukov (tu môžeme vidieť rozdiel oproti spomínaným hierarchickým metódam, kde sme vopred neurčovali počet zhlukov);
3. **Metódy Nearest Neighbour**, pri ktorých sú zhluky vytvárané z objektov, ktoré sú si navzájom najbližšími susedmi.
4. Prístupy, ktoré poskytuje **oblasť neurónových sietí** – napr. SOM a pod.

3.4.1 Algoritmus k–means

K-means je jedným z najjednoduchších a najpoužívanejších algoritmov založených na učení bez učiteľa. Procedúre predchádza zadanie množiny dát spolu s počtom zhlukov, do ktorých sa majú tieto dáta klasifikovať.

Algoritmus môžeme zhrnúť do nasledovných krokov:

1. Z priestoru klastrovaných objektov náhodne vyberieme k objektov, tie budú reprezentovať centrá klastrov.
2. Každý objekt priradíme k tomu zhluku, ktorého centrum je k nemu najbližšie.
3. Ak sú všetky objekty priradené k nejakému zhluku, je potrebné prepočítať nové centrá zhlukov (priemerom) a to tak, že vzdialenosť medzi všetkými objektmi zhluku

a novým centrom zhluku bude minimalizovaná. Napr. ak máme v 3-rozmernom priestore zhluk s dvoma bodmi $X = (x_1, x_2, x_3)$ a $Y = (y_1, y_2, y_3)$, tak centrum zhluku bude $Z = (z_1, z_2, z_3)$, kde $z_1 = (x_1 + y_1)/2$ a $z_2 = (x_2 + y_2)/2$ a $z_3 = (x_3 + y_3)/2$.

4. Opakovanie krokov 2 a 3, kým sa centrá zhlukov budú meniť.

Metóda je relatívne efektívna. Časová zložitosť je $O(tkn)$, kde n je počet vstupných objektov, k je počet zhlukov, t je počet iterácií. Väčšinou $k, t << n$. Často končí v lokálnom optime. Je vždy potrebné špecifikovať počet zhlukov, metóda je slabá pri spracovávaní zašumených údajov.

3.5 Ďalšie zhlukovacie metódy bez učiteľa

Na poli analýzy dát z DNA čipov sa využíva množstvo ďalších algoritmov. Daňou za názornosť klasického hierarchického zhlukovania je jeho prílišná časová zložitosť. Tá sa veľmi výrazne prejaví pri veľkom objeme dát (skúmanie niekoľkých tisíc génov). Vedecká verejnosť sa preto uberá k využitiu alternatívnych prístupov k tejto problematike, predovšetkým ide o neurónové siete. Algoritmus Self- Organising Maps (SOM) ako predstaviteľ nehierarchických metód využívajúcich neurónové siete zvláda veľké objemy dát a v princípe je schopný poradiť si so zašumenými vzormi dát, irelevantnými hodnotami premenných a rôznymi inými chybami, ktoré sa v reálnych dátach môžu vyskytnúť. Nevýhodou tohto algoritmu je podobne ako u k–means fakt, že počet zhlukov musí byť od začiatku pevne daný. Ďalšie problémy spomína napríklad zdroj [3]. Algoritmus Self–Organizing Tree (SOTA) vychádza z algoritmu SOM, je to však algoritmus hierarchického zhlukovania, ktorý využíva prístup zhora – dole (divisive hiearchical clustering). Obidva algoritmy majú lineárnu časovú zložitosť. Bližšie informácie spolu s porovnaním a s podrobnejšou analýzou časovej náročnosti poskytuje napríklad [7]. Z článku vyplýva, že algoritmy využívajúce neurónové siete sú naozaj vhodné na veľké množstvo dát, t.j. pri počte génov viac ako 600. V opačnom prípade totiž algoritmus klasického hierarchického zhlukovania vykazuje lepšie výsledky.

Kapitola 4

Softvér na analýzu dát z DNA čipov

Obsah tejto kapitoly bol vytvorený s pomocou zdroja [18].

4.1 Analýza obrazu

Tabuľka 4.1 uvádza základný prehľad dostupného softvéru v tejto oblasti.

Software	**Platforma**	**Typ**
AIDA Array Metrix	Windows	komerčný
ArrayPRo	Windows	komerčný
Dapple	Unix, prenositeľný kód	nekomerčný
F-scan	Unix aj Windows, implementácia v Matlabe	nekomerčný
GenePix Pro	Windows	komerčný
ImaGene	Windows, Linux a OS X	komerčný
Iconoclust	Windows	komerčný
Iplab	Windows	komerčný
P-scan	Unix aj Windows	nekomerčný
ScanAlyze	Windows	nekomerčný
Spot	Unix, Window	nekomerčný
TIGR Spotfinder	Windows, Unix	nekomerčný

Tabulka 4.1: Softvér na analýzu obrazu z DNA čipov

Software na analýzu nasnímaného obrazu DNA čipu musí vykonávať tri fundamentálne funkcie: mriežkovanie (gridding), segmentáciu a extrakciu informácií. Zatiaľ čo mriežkovanie nie je zložitým problémom, skutočnou výzvou pre používané algoritmy je segmentácia obrazu a správny odhad pozadia (background estimation). Vybavenie programov z tejto kategórie zahŕňa početné množstvo segmentačných algoritmov ako napríklad fixed circle, adaptive circle, adaptive shape a histogram. Ako algoritmy na odhad pozadia sú často používané napríklad constant background, local background a morphological opening. Každá z metód má svoje silné aj slabé stránky, ktoré je nutné pri výbere konkrétneho software zvážiť.

Niektoré komerčné softvérové balíčky obsahujú metriky na vyhodnotenie kvality jednotlivých bodov obrazu, na základe ktorých rozhodujú, ktoré body obrazu vynechať z analýzy.

Na posudzovanie slúži celý rad parametrov, ako sú priemer, plocha, odtlačok, kruhovitosť atď. Stanovenie týchto a podobných parametrov však počíta s určitou konzistenciou jednotlivých bodov v obraze DNA čipu, čo nemusí byť vždy tak. Môže sa preto stať, že sa z analýzy vyradí bod, ktorý má síce dostatočne kvalitný signál, no nespĺňa niektoré zadané kritériá. Z tohto dôvodu volá vedecká verejnosť po určitej štandardizácii v tejto oblasti a kladie si otázku, ako definovať dostatočne kvalitný bod. V tomto smere sa zdá byť plausibilná kombinácia klasického parametrizovania kvalitného bodu a technika využívaná pre program UCFS Spot.

4.2 Software na dolovanie dát

V komerčnej, ale aj v nekomerčnej sfére je v tejto oblasti pomerne bohatý výber. Existuje relatívne mnoho podobných softvérových balíčkov. Vo všeobecnosti softvér na dolovanie dát z DNA čipov zahŕňa tieto oblasti spracovania: predspracovanie a normalizáciu, zhlukovanie, klasifikáciu a vizualizáciu výsledkov. V tejto kategórii existujú štyri základné typy, ktoré sa v tejto sekcii pokúsim bližšie špecifikovať. Sú to: turnkey systémy, komplexné systémy, softvér na špecifickú analýzu a rozšírenia existujúcich programov.

4.2.1 Turnkey systémy

Turnkey systémy sú počítačové systémy, ktoré sú určené na špecifický účel. Pojem vznikol z myšlienky, že všetko, čo užívateľ potrebuje vykonať pri práci s programom, je stlačiť klávesu (turn a key). Systém je následne pripravený vykonať požadovanú operáciu. Turnkey systém na dolovanie dát z DNA čipov je teda chápaný v širšom kontexte a zahŕňa potrebné hardwarové vybavenie, serverový softvér, databázu na uloženie dát z DNA čipu, klientský a štatistický softvér. Niektoré balíky, ako napr. Genetraffic, využívajú pre jednotlivé komponenty systému open source programové vybavenie (Linux, štatistický jazyk R, PostgreSQL, Apache Web server), ostatné, ako napríklad Rossetta Resolver, využívajú proprietárne softérové technológie pre server a databázový systém, ako napríklad SunOS a Oracle.

Tabuľka 4.2 uvádza prehľad niektorých dostupných balíkov:

Software	**Platforma**	**Typ**
BASE (BioArray Software Environment)	Linux	nekomerčný
Expressionist	multiplatformný	komerčný
Genedirector	multiplatformný	komerčný
Genetraffic	Windows	komerčný
Rosetta Resolver	multiplatformný	komerčný

Tabulka 4.2: Prehľad turnkey systémov

Spomínané systémy sú väčšinou k dispozícii vo forme architektúry klient– server. Táto disponuje viacužívateľským rozhraním, čo je mimoriadne výhodné pre zdieľanie dát a výsledkov v rámci väčšej výskumnej skupiny, napríklad vo farmaceutickom priemysle. Samozrejmosťou je aj lepšia kontrola nad bezpečnosťou dát a možnosť privilegovaného prístupu. Napriek tomu, že použitie open source technológií v týchto systémoch môže významne ovplyvniť cenu takéhoto produktu, vo všeobecnosti vyžaduje pomerne veľké náklady na kúpu

a prevádzku. V menších laboratóriach a výskumných tímoch je preto inštalácia takéhoto systému skôr výnimkou.

4.2.2 Komplexné programy

Komplexné programy poskytujú metódy z rôznych fáz analýzy DNA čipov, od predspracovania dát, normalizácie, zhlukovania až po vizualizáciu. To všetko v jednom balíku. Nejde ani zďaleka o tak robustné systémy ako sú turnkey systémy, nedodávajú sa napríklad spolu s hardvérom a ani s vlastnou databázou, častokrát však disponujú rozhraním ODBC (Open DataBase Connectivity), ktoré je štandardom pre prístup do rôznych databázových systémov.

Tabuľka 4.3 uvádza prehľad dostupných komerčných aj nekomerčných komplexných balíkov.

Software	**Platforma**	**Typ**
BRB ArrayTools	Windows	nekomerčný
ArrayPRo	Windows	komerčný
Cluster	Windows	nekomerčný
Gepas	multiplatformný	nekomerčný
dChip	Windows	nekomerčný
Expression Profiler	multiplatformný	nekomerčný
GeneMaths	Windows	komerčný
GeneSpring GX	Windows	komerčný
J-Express Pro	Windows,Linux	komerčný
Partek software suites	Windows, Linux	komerčný
Spot	Unix, Windows	nekomerčný
TIGR Multiple Experiment Viewer (MeV)	Windows, Linux	nekomerčný

Tabulka 4.3: Komplexné softvérové balíky

Programy tohto typu častokrát disponujú pomerne zaujímavými užívateľskými rozhraniami. Napríklad v programe GeneSight je možné pridávať jednotlivé analytické kroky ako grafické prvky na pracovnú plochu systémom drag and drop. Takáto názorná vizualizácia zabráni nežiaducemu viacnásobnému aplikovaniu analyzačných krokov, prípadne vynechaniu niektorého z nich. Napriek nesporným výhodám použitia programov z tejto kategórie je tu aj niekoľko potenciálnych rizík. Prvým z nich je problém s nekompatibilitou dát, prípadne problémy s ich konverziou. Pokiaľ program neobsahuje určité filtre na úpravu dát od rôznych dodávateľov, užívateľ si tieto musí upraviť ručne.

Analytické spracovanie dát z DNA čipov je stále prudko rozvíjajúci sa obor. Vznikajú nové metódy a prístupy k analýze. Kým sa k aplikácii uvedie nová aktualizácia obsahujúca nejakú novú metódu, môže prejsť relatívne dosť času. Z tohto dôvodu sa vývoj niektorých komplexných programov (napr. GeneSpring alebo J–express & MAExplorer) uberá cestou zásuvných modulov. To umožňuje programátorsky zdatným užívateľom doimplementovať časť funkcionality tej ktorej aplikácie.

Treba ďalej spomenúť, že na ovládanie programov tohto typu je častokrát nevyhnutná aspoň základná znalosť štatistických metód a užívateľ si musí byť vedomý ich obmedzeniami. V opačnom prípade sa ľahko môže stať, že príde k úplne nevalídnym výsledkom a túto skutočnosť si častokrát ani neuvedomí. Toto riziko čiastočne hrozí aj pri turnkey systémoch.

4.2.3 Softvér na špecifickú analýzu

Softvér z tejto kategórie spravidla poskytuje jednu, prípadne niekoľko špecifických analýz dát, čím sa líši od all-in-one balíčkov z predchádzajúcej kategórie. Je však nutné poznamenať, že hranica medzi týmito dvoma skupinami nie je celkom jasne daná, vo všeobecnosti však platí, ze programy na špecifickú analýzu sa zaoberajú užšou problematikou a snažia sa ju spracovať čo najpodrobnejšie. Napríklad program PAM vykoná klasifikáciu nad génovými expresiami. CTWC je zasa založené na prevedení špecifického algoritmu na dátach, s programom sa komunikuje pomocou webového rozhrania. GeneCluster je určený na normalizáciu a filtráciu dát a následné zhlukovania pomocou algoritmu Self-Organizing Map (SOM). Programy na špecifickú analýzu sú väčšinou spojené s vedeckým článkom, ktorý osvetľuje štatistické a matematické pozadie metódy. Vzhľadom k tomu, že ide spravidla o špecializovaný softvér, je častokrát potrebné dáta predspracovať a aplikovať na ne ďalšie úpravy pred samotným načítaním v konkrétnom programe.

4.2.4 Rozšírenia existujúcich programov

Ako už bolo spomenuté, komplexné programové balíčky niekedy disponujú technológiou zásuvných modulov. Napríklad ArrayMiner je nová zhlukovacia utilita používajúca proprietárne zhlukovacie algoritmy a je k dispozícii jednak ako samostatná aplikácia, tak aj vo forme zásuvného modulu pre program GeneSpring. Ďalej existuje niekoľko utilít, ktoré podporujú výsledky zhlukovacích algoritmov ich vizualizáciou. Napríklad programy TreeView, Slcview a Freeview sú prehliadače dendrogramov a klastrogramov programu Cluster.

Kapitola 5

Návrh a implementácia

5.1 Úvod

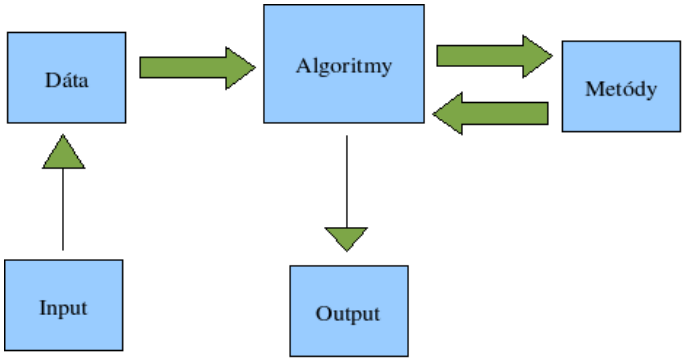
Praktická časť mojej bakalárskej práce pozostáva z implementácie konkrétnej metódy analýzy DNA čipov. Vybral som si metódu hierarchického zhlukovania, no návrh som sa snažil koncipovať tak, aby bolo možné eventuálne v budúcnosti pridávať ďalšie metódy analýzy.

5.2 Špecifikácia požiadaviek

Základnou požiadavkou je vytvorenie jednoduchej utility na zostrojenie dendrogramu klasického hierarchického zhlukovania podľa zadaných kritérií. Vstupom programu by mali byť dáta v presne špecifikovanom formáte a parametre ich spracovania. Ide predovšetkým o nastavenie metriky porovnávania génových expresií – na výber bude necentrovaný a centrovaný Pearsonov korelačný koeficient, jeho absolútna aj neabsolútna forma. Ďalej budú na výber tri metódy porovnávania zhlukov, a to Single, Complete a Average Linkage. Výstupom programu bude dendrogram vo forme XML súboru.

5.3 Základný konceptuálny návrh aplikácie

Obrázok 5.1 znázorňuje základný neformálny návrh aplikácie v podobe niekoľkých komunikujúcich modulov.



Obrázek 5.1: Štruktúra jadra aplikácie

Dáta určené k analýze sú načítané modulom IO a uložené do špeciálnych objektových typov v module DATA. Takto uložené dáta sú ďalej spracované algoritmami z modulu ALGORITMY. Tento modul obsahuje základnú štruktúru zhlukovacích algoritmov, pričom detaily a numerické operácie s dátami sú implementované v module METÓDY. Po úspešnom ukončení zhlukovacieho algoritmu sa výsledok vďaka modulu OUTPUT uloží v podobe XML do súboru. Jadro aplikácie je zastrešené modulom GUI, vďaka ktorému dostáva užívateľ rozhranie pre parametrizáciu niektorých operácií (umiestnenie vstupu a výstupu, výber konkrétnej metódy atď.). Pri návrhu bol kladený dôraz na to, aby bolo možné jednotlivé moduly ďalej rozširovať, a to napríklad pridaním ďalších algoritmov spracovania, pridaním nových metrík porovnania dát v module METÓDY, prípadne uložením výsledkov do iných (grafických) formátov.

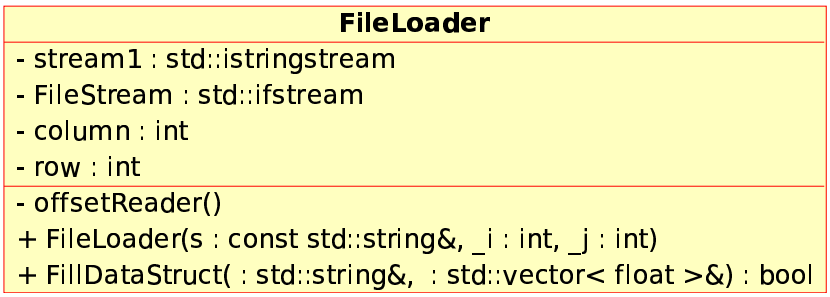
5.4 Prehľad modulov

5.4.1 Modul Input

Modul spracúva vstupné dáta. Predpokladá nasledovný formát dát.

Gén	Experiment1	Experiment2
GENE1	-0.48	0.02
GENE2	0.04	-0.01
GENE3	0.1	0.3
GENE4	0.2	-0.1

Dáta sú oddelené tabulátorom, pričom riadky zodpovedajú nameraným hodnotám génových expresií cez všetky experimenty, stĺpce sú hodnoty génových expresií všetkých génov v rámci jedného experimentu.



Obrázek 5.2: Trieda FileLoader v jazyku UML

Načítanie dát je parametrizovateľné, do konstruktoru objektu FileLoader na obrázku 5.2 je možné zadať názov súboru a údaj, od ktorého riadku, resp. stĺpca začínajú číselné dáta. Súčasná implementácia počíta so zhlukovaním génov, keďže trieda FileLoader obsahuje metódy zaručujúce naplnenie objektov z modulu DATA jednotlivými riadkami matice. Načítanie jednotlivých experimentov, teda načítanie stĺpcov matice môže byť predmetom ďalších rozšírení v budúcnosti.

5.4.2 Modul Data

Modul implementuje dátové štruktúry uschovávajúce načítané údaje. Obrázok 5.3 znázorňuje jednoduchú dedičskú hierarchiu tried implementujúcich uloženie dát. Zapojenie dedičnosti do návrhu sa javilo ako výhodná, pretože umožňuje elegantnejšiu implementáciu zhlukovacieho algoritmu.

Bázová trieda GeneAnc obsahuje chránenú dátovú zložku typu vektor reálnych čísel, ktorá uchováva dáta konkrétneho génu. Ďalej sú tu metódy implementujúce základné matematické operácie nad týmto vektorom, ako sú priemer, smerodatná odchýlka a ďalšie. V neposlednej rade je tu prítomných niekoľko virtuálnych metód, ktorých povinná implementácia v potomkoch zaručuje polymorfizmus počas behu algoritmu. Virtuálna metóda GeneAnc::compare() a jej implementácia v triedach Gene a Cluster rieši problematiku vzájomného porovnávania génov a klastrov. V obidvoch prípadoch sa tieto metódy odkazujú na metódu z modulu METODY, ktoré majú na starosti samotný výpočet korelácie. Kým však v metóde Gene::compare() ide len o porovnanie dvoch instancíi (instancie, ktorá metódu vyvolala, a odkazu na instanciu, s ktorou sa porovnáva), v prípade metódy Cluster::compare() ide o tri rôzne prípady (závisí od nastavenia dátovej zložky linkage v konstruktore triedy Cluster):

1. Single linkage – porovnávajú sa všetky objekty typu Gene patriace pod instanciu triedy Cluster, ktorá metódu vyvolala, so všetkými objektmi Gene patriacimi pod parametrom odkazovanú instanciu triedy Cluster. Metóda vráti korelačný koeficient, ktorý je spomedzi všetkých najmenší.

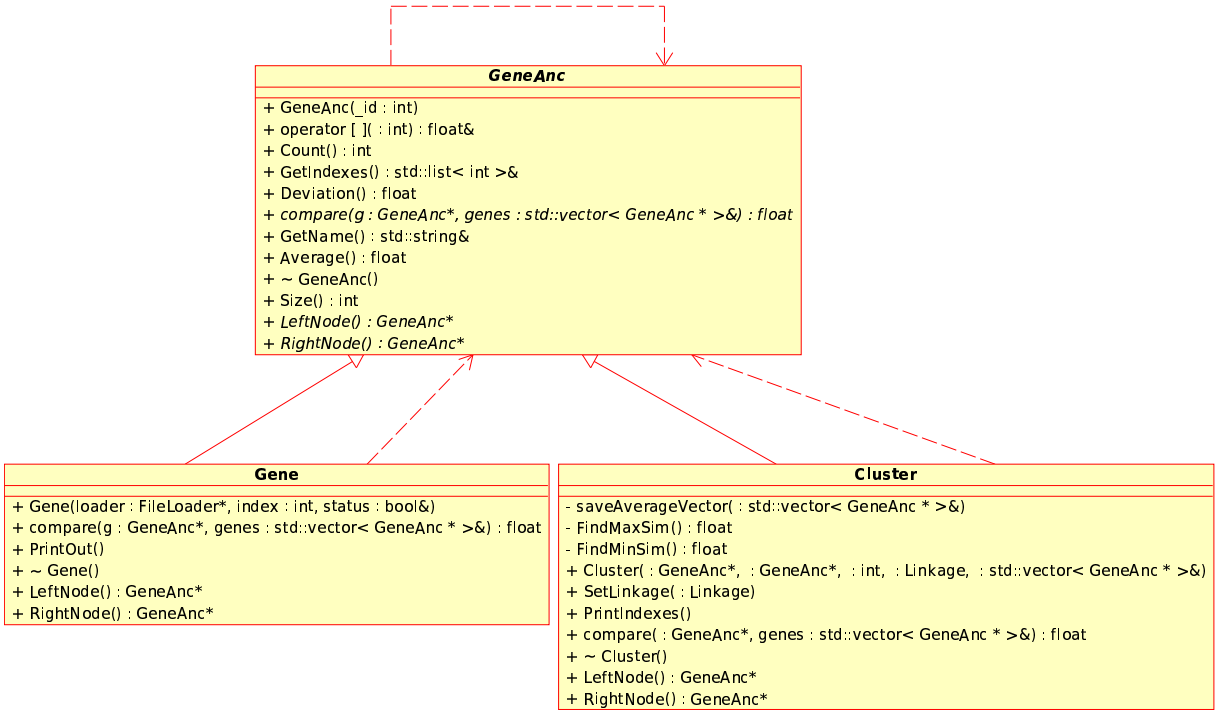
2. Complete linkage – podobne ako v predchádzajúcom bode sa porovnáva každý s každým s tým rozdielom, že metóda vráti korelačný koeficient, ktorý je spomedzi všetkých najväčší.

3. Average linkage – v prípade výberu tejto metódy sa už v konstruktore triedy Cluster nastaví vektor hodnôt pre tento klaster ako priemer hodnôt všetkých génov, ktoré tento klaster obsahuje.

5.4.3 Matica podobností

Matica podobností je pri určovaní podobností a spájaní do zhlukov kľúčovou dátovou štruktúrou. Z dôvodu ušetrenia pamäťového priestoru je táto implementovaná ako trojuholníková matica, ktorej štruktúra bola naznačená v kapitole 3.2. V prípade použitia Pearsonovho korelačného koeficientu pre porovnanie génov majú prvky na diagonále $(a_{ij}, kdei = j)$ danej matice hodnotu 1. Hľadanie maxima v matici podobností predstavuje veľmi častú operáciu, pri ktorej sa musí previesť veľký počet porovnaní (pri 100 génoch je to až 5050 porovnaní v prvom kroku zhlukovania). Preto som pristúpil k optimalizácii podľa článku [16]. Optimalizovanú maticu znázorňuje obrázok 5.4.

Ako možno vidieť z obrázka, trojuholníková matica je doplnená o vektor združujúci maximá jednotlivých stĺpcov. Ďalej sú v obrázku naznačené sivou farbou bunky korelačných koeficientov medzi i-tým zhlukom a všetkými ostatnými zhlukmi. Je zrejmé, koľko buniek vektoru je treba aktualizovať pri modifikácií/ zrušení klastru s indexom i.



Obrázek 5.3: Dedičská hierarchia

Aktualizácia matice:

Položky vektora vľavo od písmena L: bez zmeny
Položky vektora na indexe i:

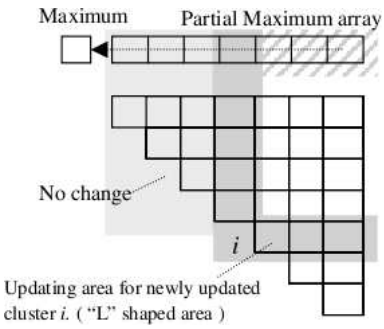
- v prípade modifikácie nastav nové maximum
- v prípade zrušenia nastav príznak zrušenia (predikát Empty())

Položky vektora vpravo od indexu i:

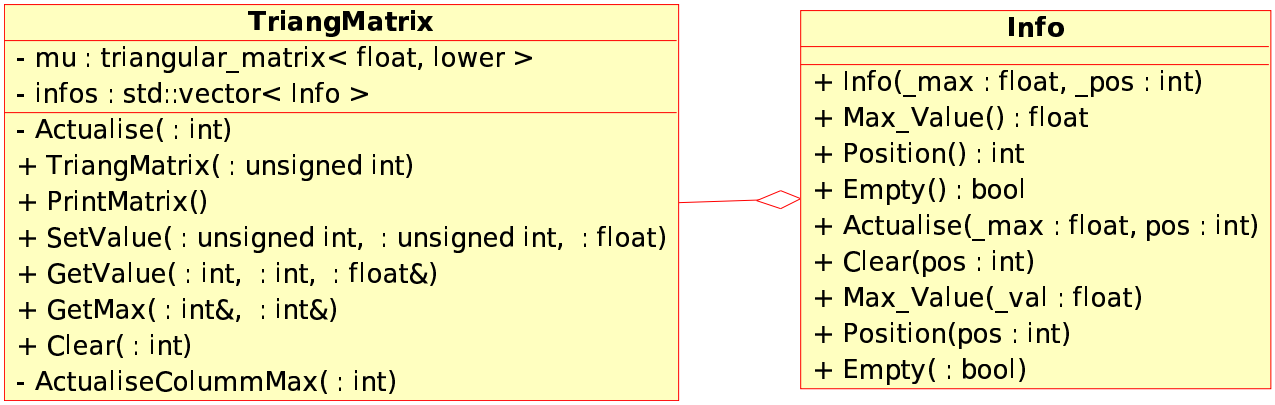
- v prípade zrušenia klastru nájdi v príslušnom stĺpci matice nové maximum a zapíš ho do vektora na pozíciu i
- v prípade modifikácie klastru:
    - pokiaľ nová hodnota >= pôvodná hodnota, zapíš novú hodnotu
    - pokiaľ nová hodnota < pôvodná hodnota
        - pokiaľ bolo pôvodné maximum práve z i-teho klastru, nájdi v príslušnom stĺpci nové maximum

Obrázok 5.5 znázorňuje návrh matice podobností aj s uvedenou optimalizáciou v jazyku UML.

Z diagramu vidieť, že trieda TriangMatrix obsahuje súkromné dátové zložky mu a infos. Zložka mu je špecializáciou šablóny `triangular_matrix` z použitej knižnice Boost (`www.boost.org`), ktorá implementuje trojuholníkovu maticu a základné operácie nad ňou. Zložka infos je špecializáciou šablóny `vector` z STL, ktorá je naplnená štruktúrou Info. Táto štruktúra implementuje práve jednu položku vektora maxím trojuholníkovej matice a obsahuje zložky a prístupové metódy k informáciam o konkrétnej položke. Pri vzniku instancie triedy TriangMatrix sa inicializuje zároveň trojuholníková matica ako aj vektor



Obrázek 5.4: Optimalizovaná matica podobnosti prevzatá z [16]



Obrázek 5.5: UML návrh matice podobností

hodnotou zodpovedajúcou počtom génov v skúmaných dátach. V triede TriangMatrix je ďalej implementovaných niekoľko metód na aktualizáciu matice, ktoré zapúzdrujú operácie nad jej zložkou `mu`. Treba poznamenať, že trieda nedisponuje žiadnou vstavanou inteligenciou, o jednotlivých operáciach nad instanciou triedy TriangMatrix rozhoduje samotný riadiaci algoritmus (viď dalej).

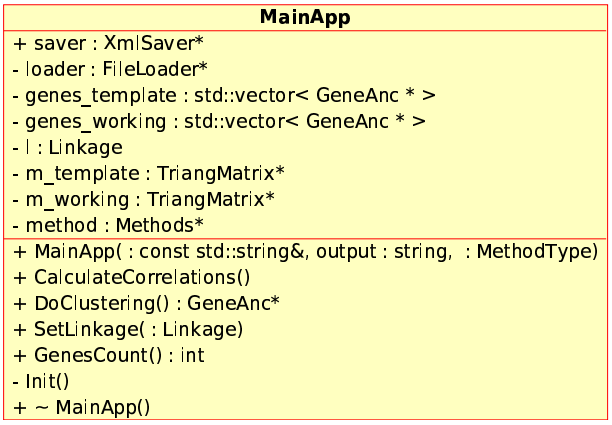
5.4.4 Algoritmus

Ako už bolo spomenuté v úvode tejto kapitoly, zameral som sa na algoritmus hierarchického zhlukovania. Súčasná implementácia zahŕňa využitie Pearsonovho korelačného koeficientu pri porovnávaní jednotlivých génov a tri rôzne metriky spájania zhlukov – single, complete a average linkage clustering. Jednotlivé kroky algoritmu zapúzdruje trieda MainApp zobrazená na obrázku 5.6. Postup činnosti algoritmu je možné zhrnúť nasledovným pseudokódom:

Riadiaci algoritmus zhlukovania

while (nie sú načítané všetky dáta)
    načítaj informácie o géne a číselný vektor do novej instancie
    triedy Gene a tú ulož do vektoru ukazateľov na typ GeneAnc (ďalej GeneAncVec)
Inicializuj instanciu triedy TriangMatrix (ďalej len Matrix) s veľkosťou

zhodnou s počtom položiek v GeneAncVec.
Porovnaj každú položku GeneAncVec s ostatnými v GeneAncVec pomocou
metódy Gene::compare() a výsledky zapíš do Matrix
Vytvor si kópiu `Matrix => Matrix_cpy`
Vytvor si kópiu `GeneAncVec => GeneAncVec_cpy`
while (`Matrix_cpy` obsahuje neprázdne `indexy => predikát Empty()` vracia false)
    vráť indexy génov, resp. klastrov, medzi ktorými je hodnota
    korelačného koeficientu najväčšia
    vytvor novú instanciu triedy Cluster, ktorej nastavíš
    index na hodnotu väčšieho z nájdených indexov z predošlého kroku,
    nastav ukazatele tejto instancie, aby ukazovali na objekty, ktorých indexy
    boli nájdené v predchádzajúcom kroku
    objektu s menším indexom nastav v Matrix príznak Empty() na true
    Ulož si odkaz na vytvorený cluster do pomocného ukazateľa



Obrázek 5.6: Trieda MainApp v jazyku UML

Pokiaľ všetko prebehlo korektne, mal by byť po ukončení cyklu k dispozícii pomocný ukazateľ odkazujúci na koreň dendrogramu previazaného ukazateľmi v pamäti. Indexu, ktorým sa nastaví nový zhluk pri spájaní dvoch objektov nie je vybraný na maximum týchto prvkov náhodne. V konečnom dôsledku vďaka tomu totiž ostane po skončení algoritmu práve jeden zhluk s indexom rovnajúcim sa prvku s maximálnym indexom v celom vektore génov. To je výhodné, pretože sa tým vyhnem mazaniu záznamu (nastavovaniu predikátu Empty()) o poslednom prvku z matice podobností, presnejšie z vektora združujúceho záznamy o nej. Tento krok by totiž vzhľadom k jej konštrukcii činil určité implementačné komplikácie.

5.4.5 Metódy

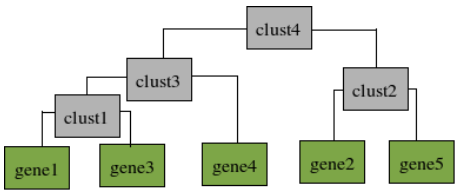
Modul metódy zahŕňa triedu Methods. Tá slúži na zapúzdrenie metrík určovania podobnosti. Sú implementované ako statické metódy a rozhranie instancie tejto triedy vracia ukazateľ na príslušnú statickú metódu. Aktuálne je implementovaný výpočet Pearsonovho korelačného koeficientu, no trieda je navrhnutá tak, aby bolo možné jednoducho pridávať ďalšie metriky určovania podobnosti.

5.4.6 Výstup XML

V situácii, keď je dendrogram vybudovaný v pamäti, je potrebné ho v prehľadnej štruktúrovanej podobe zobraziť. K tomuto účelu je vhodné napríklad XML. Na zápis do formátu XML bola použitá knižnica TinyXml (`http://www.grinninglizard.com/tinyxml/`). Pri zápise do XML bolo potrebné systematicky prehľadávať jednotlivé uzly stromu a zapísať ich názov. Pre priechod dendrogramom som si zvolil algoritmus Pre Order, ktorý má s využitím rekurzie nasledovnú podobu:

1. Zapíš aktuálny uzol
2. Prechádzaj ľavý podstrom
3. Prechádzaj pravý podstrom

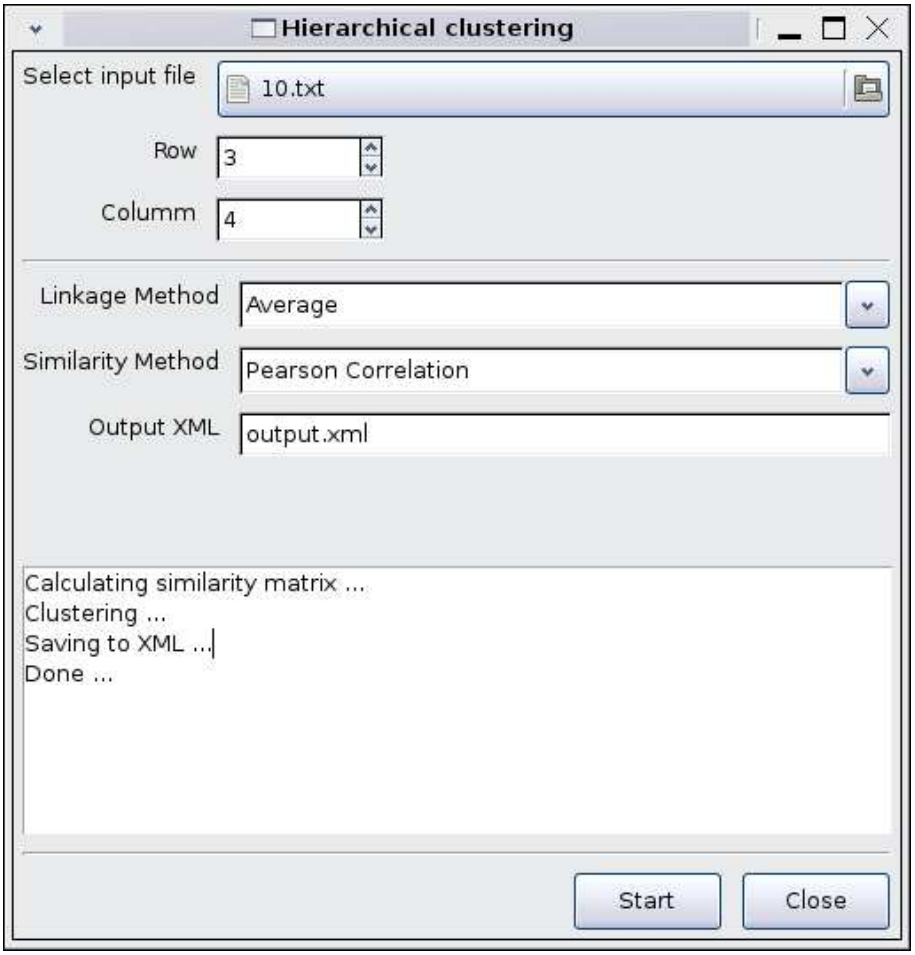
Dendrogram sa zvyčajne vyskytuje vo vizualizovanej podobe, na obrázku je znázornený ten istý dendrogram v dvoch rôznych formách zápisu. Zhluky v XML sú označené ako nodeX, kde X udáva poradie, v ktorom bol vytvorený. Nižšie číslo teda znamená väčšiu podobnosť.




```
<node4>
  <node3>
    <node1>
      <gene1/>
      <gene3/>
    </node1>
    <gene4/>
  </node3>
  <node2>
    <gene2/>
    <gene5/>
  </node2>
</node4>
```

5.4.7 GUI

Na grafické užívateľské rozhranie nebol v priebehu návrhu ani implementácie kladený dôraz. Ide skôr o doplnok, ktorý užívateľovi poskytuje väčší komfort pri výbere konkrétnych metód a zároveň informuje užívateľa o práve prebiehajúcich operáciach. Grafické užívateľské rozhranie bolo realizované využitím knižnice wxWidgets 2.8 (`http://www.wxwidgets.org/`) a programu wxFormBuilder.(`http://wxformbuilder.org/`). Aplikácia umožňuje vytvárať grafické užívateľské rozhranie štýlom WYSIWYG, čo prácu značne urýchľuje. Je schopná vygenerovať potrebný kód týkajúci sa GUI. Následne je potrebné už len dodať funkcionalitu jednotlivým udalostiam aplikácie. V tomto prípade ide predovšetkým o tlačidlo Start, po ktorého stisknutí sa inicializuje instancia triedy MainApp na parametre, ktoré zadal užívateľ pomocou GUI. Obrázok 5.7 poskytuje náhľad na vytvorené grafické užívateľské rozhranie.



Obrázek 5.7: Ukážka grafického užívateľského rozhrania

Kapitola 6

Testovanie a porovnanie výsledkov

6.1 Základné informácie o testovacích dátach

Moja implementácia je plne kompatibilná s dátami z projektu Lymphoma/Leukemia Molecular Profiling Project[11]. Tento projekt sa zaoberá skúmaním leukémie B-lymfocytov. Pre tieto účely bol navrhnutý komplementárny DNA čip, tzv. Lymphochip. Obsahuje výber jednak takých génov, ktorých génová expresia je v lymfatických bunkách výrazná, a ďalej gény, ktoré pravdepodobne hrajú úlohu pri vzniku rakoviny. Genetické informácie boli spracované v súlade s postupom, ktorý som uviedol v kapitole 2. Dáta boli potom upravené konvenčnými metódami, napríklad úpravou merítka a filtrovaním. Po týchto úpravách sú dáta v stave vhodnom na analýzu zhlukovacími algoritmami.

6.2 Porovnávaný softvér

Po počiatočnom uvážení som sa rozhodol porovnávať výstupy mnou vytvoreného programu s komplexným softvérovým balíkom MeV. Tento softvér poskytuje nepreberné množstvo rôznych analytických algoritmov vrátane predspracovania dát.

6.3 Testovanie

V prvej fáze testovania som sa rozhodol pre malé množstvo dát, aby bolo možné výstupy oboch programov porovnať presne, gén po géne. Zhlukovanie u oboch programov bolo spustené s nastavením ako udáva tabuľka:

Vstupný súbor	first10.txt
Pozícia začiatku dát	3.riadok, 4. stĺpec
Metóda určovania vzdialenosti	Pearsonov korelačný koeficient (centrovaný)
Linkage metóda	Average–Linkage
Výtupný súbor	first10.xml

Následne bol vizuálne porovnaný výstup first10.xml s programom MeV. Štruktúra oboch výstupov nebola identická. Ako názornú ukážku uvádzam porovnanie výstupu môjho programu a výsledok z programu MeV prepísaného do formátu XML.

```
<node7>
    <node5>
        <GENE1835X />
        <GENE1836X />
    </node5>
    <node6>
        <GENE1380X />
        <node4>
            <GENE1933X />
            <node3>
                <GENE1865X />
                <node2>
                    <node1>
                        <GENE1932X />
                        <GENE1931X />
                    </node1>
                    <GENE1930X />
                </node2>
            </node3>
        </node4>
    </node6>
</node7>
```

```
<node7>
    <node4>
        <GENE1835X />
        <node3>
            <GENE1836X />
            <GENE1865X />
        </node3>
    </node4>
  <node6>
    <GENE1380X />
    <node5>
        <GENE1933X />
        <node2>
            <node1>
                <GENE1932X />
                <GENE1931X />
            </node1>
            <GENE1930X />
        </node2>
    </node5>
  </node6>
</node7>
```

Je vidieť, že že prvé dva zhluky sa v obidvoch prípadoch vytvorili v rovnakom poradí a s identickým obsahom. Ďalej však dochádza k určitým rozdielom. Zatiaľ čo v mojom programe sa v ďalšom zhluku priradil prvok GENE1865X, v referenčnom programe sa vytvoril nový zhluk s génmi GENE1836X a GENE1865X. V mojom programe sa spomínaný zhluk vytvoril až v piatom kroku, pričom sa GENE1836X spojil s GENE1835X. GENE1835X bol však u mňa spojený o jeden krok neskôr. Uvedené rozdiely môžu byť spôsobené napríklad použitím čísel s pohyblivou desatinnou čiarkou s inou presnosťou, ktorá sa prejaví pri porovnávaní prvkov matice podobností.

Podobne boli analyzované aj ďalšie implementované metódy, teda Single a Complete Linkage s využitím centrovaného aj necentrovaného Pearsonovho korelačného koeficientu. Možno skonštatovať, že výstupy neboli identické, väčšinou však išlo o rozdiely, ktoré typovo zodpovedajú uvedenému príkladu.

6.3.1 Testovanie väčšieho objemu dát

Cieľom testovania na väčšom objeme dát už nebolo overiť správnu funkčnosť algoritmu, skôr jeho výkonnosť. Ako už bolo spomenuté, hierarchické zhlukovanie patrí medzi časovo náročné metódy. Prejavilo sa to aj v tomto prípade, keď pri počte génov cca 4000 a metóde average linkage sa pohybovala práca algoritmu v časovom horizonte okolo päť minút. V porovnaní s ostatným dostupným nekomerčným softvérom však v tejto kategórii obstál pomerne dobre (napríklad program Cluster bol na tom výkonnostne horšie, a to taktiež neposkytuje priamo grafický výstup).

Kapitola 7

Záver

Táto práca sa pokúša poskytnúť ucelený prehľad spracovania komplementárnych DNA čipov – od ich fyzickej realizácie, cez analýzu dát až po spracovanie výsledkov. Ide o veľmi rozsiahlu multiodborovú problematiku, ktorá navyše v súčasnosti prežíva skutočný rozkvet. Trendy, ktoré sú zaužívané dnes, nahradia zajtra progresívnejšie alternatívy. Sledovanie tohto vývoja je určite otázkou na dlhšie časové obdobie.

Práca mi priniesla možnosť prakticky si vyskúšať konkrétnu etapu spracovania dát z DNA čipu. Vytvoril som fungujúcu aplikáciu, ktorú možno zaradiť medzi softvér na špecifickú analýzu. Vzhľadom k tomu, že som sa snažil moju implementáciu zvolených algoritmov navrhnúť s dôrazom na to, aby mohla byť v budúcnosti rozšírená, predpokladám doimplementovanie ďalších metód. Zaujímavou a progresívnou oblasťou sú práve algoritmy využívajúce neurónové siete. Inšpiráciou môže byť napríklad aplikácia MeV, ktorá poskytuje celý rad metód z tejto oblasti.

Vzhľadom k získaným výsledkom by ďalej bolo vhodné automatizovať proces porovnávania výstupov programu s niektorou vedecky uznávanou aplikáciou. Predovšetkým pri väčšom objeme dát by tak bolo možné presne vyhodnotiť, do akej miery bol program úspešný vzhľadom k referenčnej aplikácii, a to aj na väčšom objeme dát. S tým je spojená prirodzene aj určitá úprava, prípadne oprava už existujúceho kódu s cieľom viac sa priblížiť referenčným výsledkom.

Súčasný výstup mojej implementácie vo formáte XML sa javil už počas testovania ako nepraktický. Pri väčšom množstve dát je totiž ich vizuálne skúmanie pomerne obtiažne. V budúcnosti by teda modul výstupu XML mohol byť nahradený iným. Napríklad takým, ktorý by poskytoval grafický výstup podobne, ako je to v programe MeV.

Literatura

[1] Gert Thijs a kol. *Genomics and Proteomics Engineering in Medicine and Biology.* Addison-Wesley Publishing Company, 1996. ISBN 0-201-13447-0.

[2] J. Buhler. Anatomy of comparative gene expression study. `http://www.cs.wustl.edu/ jbuhler/research/array/`.

[3] Joaquín Dopazo. *Methods of Microarray Data Analysis II.* Springer US, 2002. ISBN 978-0-306-47598-6.

[4] Michael Eisen. Cluster and treeview manual. `http://rana.lbl.gov/manuals/ClusterTreeView.pdf`.

[5] Miroslav Hrstka. *Obecná biologie.* VUT v Brně, Fakulta chemická, 2005. ISBN 80-214-3057-5.

[6] A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: a review. *ACM Computing Surveys*, 31(3):264–323, 1999.

[7] Alfonso Valencia Javier Herrero and Joaquín Dopazo. A hierarchical unsupervised growing neural network for clustering gene expression patterns. pages 126–136, 2000.

[8] Jozef Kriška. Hierarchická a nehierarchická zhluková analýza. `http://www2.fiit.stuba.sk/ kapustik/ZS/Clanky0607/kriska/index.html`.

[9] WWW stránky. Analysis of multiple experiments. `http://www.tm4.org/documentation`.

[10] WWW stránky. Hierarchical clustering algorithms. `http://home.dei.polimi.it/matteucc/Clustering`.

[11] WWW stránky. Lymphoma/leukemia molecular profiling project. `http://llmpp.nih.gov/lymphoma/`.

[12] WWW stránky. National human genome research institute. `http://www.genome.gov`.

[13] WWW stránky. Overview of genechip technology. `http://www.ohsu.edu/gmsr/amc/amc_technology.html`.

[14] WWW stránky. Pearson's correlation. `http://davidmlane.com/hyperstat/A34739.html`.

[15] WWW stránky. Practical microarray analysis. `http://mki0252c.medinn.med.uni-muenchen.de`.

[16] Taek-Soo Kim Sung Young Jung. An agglomerative hierarchical clustering using partial maximum array and incremental similarity computation method. In *ICDM archive Proceedings of the 2001 IEEE International Conference on Data Mining*, pages 265–272. IEEE Computer Society Washington, DC, USA, 2001. ISBN 0-7695-1119-8.

[17] Kardi Teknomo. Similarity measurement. `http://people.revoledu.com/kardi/tutorial/similarity/`.

[18] Dennis Shun Chiu Lam Yuk Fai Leung and Chi Pui Pang. *A Practical Approach to Microarray Data Analysis*. Springer US, 2003. ISBN 978-0-306-47815-4.