VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

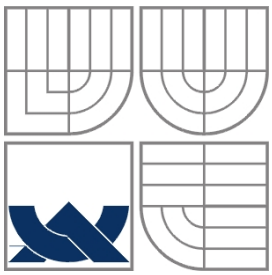
FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

APLIKACE PRO ZPRACOVÁNÍ DAT Z OBLASTI GENOVÉHO INŽENÝRSTVÍ

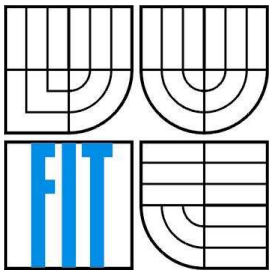
DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE Bc. JAN BRYCHTA
AUTHOR

BRNO 2008



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

APLIKACE PRO ZPRACOVÁNÍ DAT Z OBLASTI GENOVÉHO INŽENÝRSTVÍ

APPLICATION FOR THE DATA PROCESSING IN THE AREA OF GENOME ENGINEERING

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

Bc. JAN BRYCHTA

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. PAVEL OČENÁŠEK

BRNO 2008

Abstrakt

Tato práce má několik cílů. Jedním z nich je seznámit s problematikou genového inženýrství, zejména pak s fragmentací DNA, makromolekulou DNA, metodami pro purifikaci a separaci nukleových kyselin, enzymy používanými k úpravám těchto kyselin, amplifikací a také seznámit se shlukovou a gradientovou analýzou. Dalším cílem je pak prostudovat existující nástroj a srovnat ho s návrhem vlastní aplikace, což si klade jako cíl třetí. S návrhem úzce souvisí poslední z cílů. Je jím samotná implementace a zápis, jak byla aplikace otestována s reálnými daty. Získané výsledky budou diskutovány, stejně jako možnosti dalšího rozšíření.

Klíčová slova

DNA, fragmentace DNA, Syntax 2000, genové inženýrství, amplifikace, purifikace, elektroforéza, shluková analýza, gradientová analýza, koeficient podobnosti, kontingentní tabulka, UPGMA, PCA, PCoA, ADO.NET, DataSet.

Abstract

This master's thesis has a few objectives. One of them is to acquaint with the problems of genome engineering, especially with fragmentation of DNA, the macromolecule DNA, the methods for purification and separation of the nucleic acids, the enzymes used for modification of these acids, amplification and get to know with cluster and gradient analysis as well. The next aim is to peruse the existed application and compare it to the layout of the proposed application, that is the third aim. The last one from the objectives is the implementation and the report how was the application tested by the real data. The results will be discussed as well as the possibilities of the further extension.

Keywords

DNA, fragmentation of DNA, Syntax 2000, genome engineering, amplification, purification, electrophoresis, cluster analysis, gradient analysis, similarity coefficient, associative table, UPGMA, PCA, PcoA, ADO.NET, DataSet.

Citace

Jan Brychta: Aplikace pro zpracování dat z oblasti genového inženýrství, diplomová práce, Brno, FIT VUT v Brně, 2008

Aplikace pro zpracování dat z oblasti genového inženýrství

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně pod vedením Ing. Pavla Očenáška. Další informace mi poskytl konzultant Mgr. František Zedek z Ústavu botaniky a zoologie Přírodovědecké fakulty Masarykovy university.
Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

……
Jan Brychta
16. května 2008

Poděkování

Rád bych zde poděkoval svému vedoucímu diplomové práce Ing. Pavlu Očenáškovi za vedení a odbornou pomoc. Děkuji také konzultantovi Mgr. Františku Zedkovi za odbornou pomoc v oblasti botaniky a vlastního genetického inženýrství.

Obsah

1 Úvod

Genové inženýrství je jednou z velmi zajímavých oblastí lidského bádání. Jako každá jiná vědní disciplína se neobejde bez použití počítačových aplikací, které jí pomáhají najít řešení různých výpočetních problémů. Jde především o získávání a analýzu genomických dat, což jsou v našem případě sekvence nukleových kyselin (mohly by to být sekvence bílkovin apod.), ukládání, praktické uchovávání a grafické zobrazení těchto dat.

V následujících kapitolách se budeme soustředit na elektrolýzu a amplifikaci nukleových kyselin, manipulaci s nukleovými kyselinami, shlukovou a gradientovou analýzu, existující nástroje, zanalyzování a navržení aplikace poskytující řešení některých vybraných numerických metod, její implementaci a testy.

V první kapitole se seznámíme s makromolekulou DNA, která je nositelkou genetické informace všech buněčných organismů. Obecně si DNA popíšeme, podíváme se na její strukturu, vlastnosti, význam a její získávání z organismů.

Druhá kapitola popisuje elektroforézu, která patří v molekulární biologii k nejpoužívanějším separačním technikám při izolaci a analýze nukleových kyselin a bílkovin. Zejména půjde o gelovou elektroforézu.

Třetí kapitola podává popis manipulace s nukleovými kyselinami. Jsou zde zmíněny enzymy, které se používají k úpravám nukleových kyselin. Rozdělíme si enzymy podle substrátové specifity a podle typu reakcí.

Další kapitola rozebere význam polymerázové řetězové reakce (PCR), která zastupuje nejvýznamněji amplifikaci nukleových kyselin.

V páté kapitole si ukážeme shlukovou a gradientovou analýzu, koeficienty statistických metod, dle kterých se vyhodnocují míry podobnosti živočišných druhů, či nejzákladnější aglomerativní algoritmy.

Další kapitola uvede existující počítačové nástroje používané při zkoumaných metodách. Jde především o program Syntax, jenž má v oblasti genového inženýrství hojné uplatnění.

V sedmé kapitole zanalyzujeme a navrhneme desktopovou aplikaci, která bude poskytovat řešení těch numerických metod, které jsou pro nás významné.

Osmá kapitola pojednává o samotné implementaci. Je zde obsaženo, v jakém prostředí byla aplikace vytvořena a za pomoci jakých nástrojů a technik. V této souvislosti budou čtenáři přiblíženy hlavní pojmy spojené s vytvářením desktopových aplikací pracujících s databázovými daty.

Devátá kapitola uvádí vybrané testy. Tyto testy věrně simulují reálné prostředí a běžná data, která budou botaniky používána.

V závěru práce shrneme doposud prostudovanou problematiku a nastíníme další možný vývoj aplikace.

2 DNA

Celou tuto kapitolu budeme věnovat DNA. Objasníme si pojem DNA, kde se s ním můžeme setkat, dále si popíšeme strukturu DNA, její vlastnosti a význam. Na závěr kapitoly si vysvětlíme, jak se získává DNA z organismů.

2.1 Obecný popis DNA

DNA [1] je nositelkou genetické informace všech organismů. Je pro život nezbytnou látkou, která ve své struktuře kóduje program života. Jsou zde kódovány plány pro syntézu nukleových kyselin a bílkovin, které pak ovlivňují vývoj a vlastnosti každé buňky žijícího organismu a tím zároveň kompletně vlastnosti celého organismu.

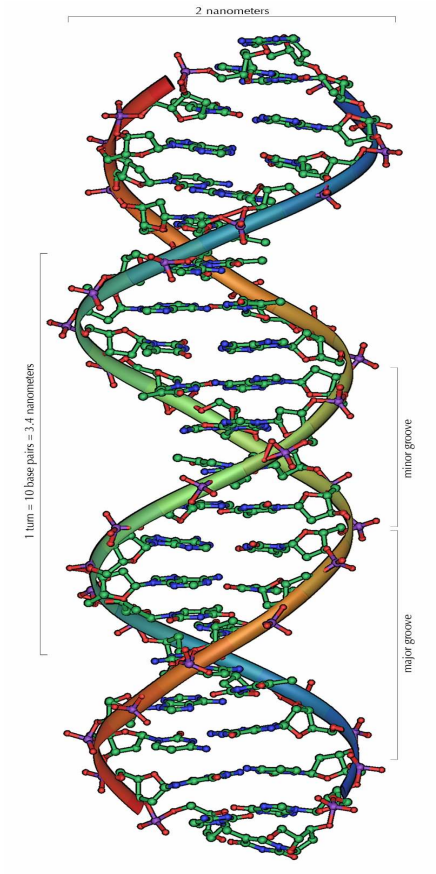
„Mezi jedinci jednoho živočišného či rostlinného druhu nalezneme v DNA sekvenci rozdíly. Na druhou stranu prakticky všechny buňky, bez ohledu na to, z jaké části organismu pocházejí, nesou úplnou dědičnou informaci pro vývoj celého organismu – všechny mají stejnou DNA. Nicméně jen určitá část této informace je v konkrétní buňce realizována. Tím se od sebe buňky navzájem v průběhu růstu diferencují. Buňky lidského těla tak tvoří různé tkáně, například játra, kůži, svaly apod. Pro každou konkrétní buňku je DNA určitou kuchařkou, podle níž realizuje svůj specifický program.“ [1, str.12]

DNA řídí a udržuje při životě celý organismus vydáváním pokynů buňce pro vytváření základních molekul bílkovin. Bílkoviny mají velmi různorodé vlastnosti a funkce. Zmiňme např. bílkoviny stavební (kosti, svaly) nebo regulační (enzymy). Úsek sekvence DNA [1] pro kódování jedné bílkoviny se nazývá gen. Každý gen může mít jednu nebo několik forem – alel. U prokaryotních organismů (bakterie) se genetická informace nachází na tzv. prokaryotním chromozomu a v plazmidech.

2.2 Struktura DNA

„DNA je biologická makromolekula skládající se ze dvou dlouhých řetězců zkroucených do dvojité šroubovice tvořící kroucený žebřík (obr. 2.1). V každém řetězci se střídá pětiuhlíkový cukr deoxyribóza se zbytkem kyseliny fosforečné (tzv. fosfátem). Na deoxyribózu je z vnitřní strany dvojšroubovice připojena heterocyklická sloučenina, tzv. báze. Na místě báze se mohou v DNA vyskytnout čtyři různé heterocyklické sloučeniny. Dvě z nich jsou odvozeny od struktury purinu a dvě od struktury pyrimidinu. Mezi purinové báze patří Adenin (A) a Guanin (G), mezi pyrimidinové patří Tymim (T) a Cytosin (C). Báze do sebe specificky zapadají a tvoří tak vazebné páry, které při sobě

drží pomocí vodíkových můstků a spojují tak oba řetězce molekuly DNA do dvojšroubovice.“ [1, str.12]



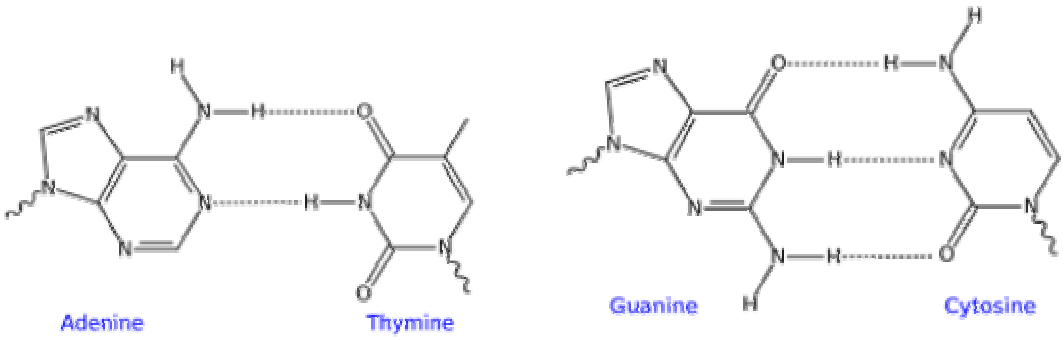
Obr. 2.1 Dvojšroubovice DNA [3]

Dvojřetězcová struktura [1] DNA je výhodná pro uchování genetické informace a lépe odolává vnějšímu poškození. Genetická informace je v DNA zapsána v pořadí bází. Tyto báze se mohou pojit pouze následujícím způsobem: A se pojí s T a C se pojí s G (obr. 2.2).

„*Trojice deoxyribóza + zbytek kyseliny fosforečné + báze, tzv. nukleotid, tvoří základní stavební jednotku (monomer) molekuly DNA. Protože na místě báze se mohou vyskytovat čtyři různé molekuly, odvozujeme pro nukleotid čtyři různé názvy podle obsažené báze: deoxyadenosinfosfát, deoxytymidinfosfát, deoxyguanosinfosfát a deoxycytidinfosfát. Pokud z trojice látek, ze kterých je složen nukleotid, odebereme zbytek kyseliny fosforečné, dostaneme tzv. nukleosid.*

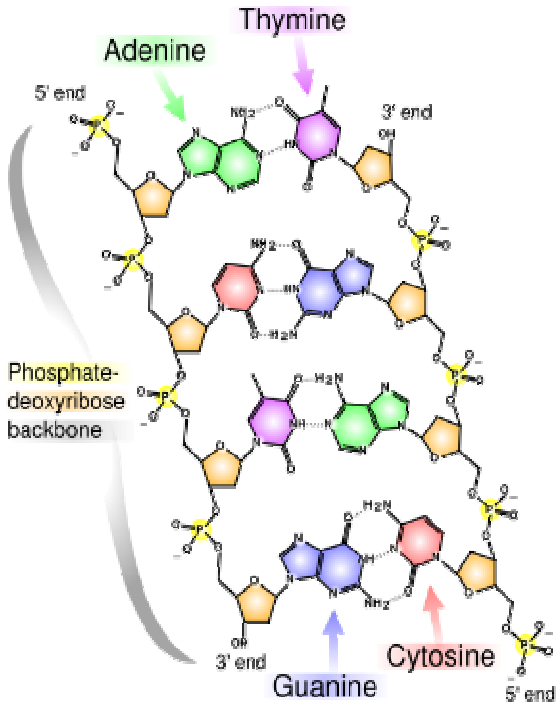
Pro zopakování ještě uveďme, že hřbet dvojšroubovice je tedy tvořen cukrem a fosfátem a vnitřek nukleovou kyselinou (nukleovými bázemi).

- *Báze: adenin (A), tymin (T), guanin (G), cytosin (C)*
- *Nukleosid: báze + cukr (deoxyribosa)*
- *Nukleotid: báze + cukr + zbytek kyseliny fosforečné*“ [1, str.13]



Obr. 2.2 Vazby mezi bázemi [3]

V nukleotidu má každý atom své číslo. Uhlík v cukerné složce, na němž je navázán zbytek kyseliny fosforečné, má číslo 5´, uhlík, na němž je navázána skupina OH, má číslo 3´. Každé vlákno, řetězec, má podle konvence 3´-hydroxylový a 5´-fosfátový konec. V dvojšroubovici mají vlákna opačný směr, tj. jedno vlákno je orientováno z konce 3´ ke konci 5´ a druhé vlákno z konce 5´ ke konci 3´. Obě vlákna ve dvojšroubovici DNA jsou tedy vůči sobě komplementární, ale také opačně orientovaná (obr. 2.3).



Obr. 2.3 Chemická struktura DNA [3]

„Podíváme-li se pozorně na schéma dvojřetězové molekuly DNA, můžeme si všimnout, že deoxyribózy jsou v protějších řetězcích orientovány opačnými směry. Směřuje-li v horním řetězci kyslík ve vrcholu pětiuhlíkového cyklu deoxyribózy vždy doleva, pak v dolním řetězci směřuje vždy

doprava. Každý z řetězců tedy má svůj směr. Toho právě při čtení genetické informace využívá enzym RNA polymeráza. Kvůli této orientaci vláken se RNA polymeráza může po konkrétním řetězci vydat pouze jedním směrem, který je dán orientací 5´- a 3´- konce.“ [1, str.14]

Z tohoto faktu vychází i základní konvence, kterou se řídí zápisy jakékoli sekvence bází v nukleových kyselinách v učebnicích a vědeckých textech – uváděné pořadí vždy představuje sled nukleotidů od 5´konce vlákna ke 3´ konci vlákna.

„Nyní se ještě podívejme na možné nahlížení na strukturu DNA. U nukleových kyselin rozlišujeme primární, sekundární, a terciální strukturu.

Primární struktura

Primární struktura je dána pořadím nukleotidů v sekvenci. Nese v sobě genetickou informaci.

Sekundární struktura

Forma stočení dvojšroubovice. Vlákna DNA se přirozeně stáčí do dvojšroubovice, avšak forma stočení není vždy za všech podmínek stejná. Může se vyskytovat v několika formách, které mají vliv na reaktivitu molekuly:

- *ds forma A – Pravotočivá; 10 párů bází na otáčku; průměr vlákna je 2,3 nm*
- *ds forma B – Pravotočivá; 11 párů bází na otáčku; průměr vlákna je 1,9 nm*
- *ds forma Z – Levotočivá; 12 párů bází na otáčku; průměr vlákna je 1,8 nm*

Terciální struktura

Velmi dlouhá molekula DNA není jen neuspořádaným klubkem náhodně zamotaného vlákna. Celá molekula se velmi pečlivě několikanásobně navíjí a skládá.

Pro lepší představu o velikosti makromolekuly DNA si uveďme příklad. Pokud by se nám podařilo rozmotat a natáhnout celou molekulu DNA z jedné lidské buňky, měřila by téměř dva metry. A kdybychom ji zvětšili do tloušťky niti, byla by dlouhá asi 300 kilometrů.“ [1, str.14-15]

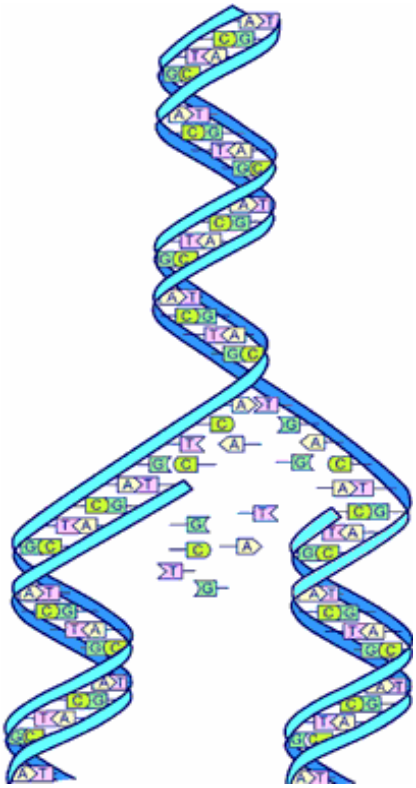
2.3 Vlastnosti a význam DNA

Základní vlastní úlohou DNA je uchovávat a dále předávat do dalších generací genetickou informaci, která je zapsaná v její primární struktuře. Tato informace se přenáší (dědí) z generace na generaci. Člověk obdrží od každého rodiče 23 chromozomů. Prvním a zásadním krokem při uchovávání dědičné informace a při jejím přenosu z rodičů na potomky je její zdvojení.

2.3.1 Replikace DNA

„Základní mechanismus replikace, tedy zdvojení deoxyribonukleové kyseliny je znám již od objevu struktury její molekuly začátkem padesátých let. Vlákna DNA jsou složitým pochodem za pomoci řady proteinů rozplétána a k uvolněným vláknům jsou prostřednictvím enzymů dosyntetizovány komplementární protějšky (obr. 2.4). Tak z jedné dvojvláknové molekuly DNA vznikají dvě identické dvojvláknové molekuly. Při dělení buňky pak každá z nich přechází do jedné buňky dceřiné. Tento proces musí proběhnout při každém dělení buňky.“ [1, str.15]

Vlastní syntézu vlákna [1] DNA řídí enzym DNA-polymeráza, který katalyzuje přenos nukleotidů na rostoucí řetězec. Protože je bytostně důležité, aby kopie byly dokonale stejné, má DNA-polymeráza schopnost odstranit připojený nukleotid, pokud není připojen komplementárně. Tento proces se pak označuje jako reparace DNA.



Obr. 2.4 Replikace DNA [3]

2.4 Získávání DNA z organismů

Objasněme si, jak se DNA získává z organismů a jaké kroky je nutné udělat, aby se genetická informace uložená v podobě sekvence nukleotidů převedla do podoby textové sekvence zapsané kombinací čtyř písmen.

Izolace nukleových kyselin

„Podstatou izolace nukleových kyselin je oddělení těchto kyselin od ostatního biologického materiálu, kterým jsou nukleové kyseliny obklopeny. Po izolaci nukleových kyselin je nutné ještě oddělit od sebe jednotlivé nukleové kyseliny. Pokud je například cílem izolovat molekulu DNA, musíme ještě z nukleových kyselin, které si jsou svojí chemickou stavbou velmi podobné, vybrat pouze a právě molekulu DNA.“ [1, str.18]

Štípání molekuly DNA

„Pokud je DNA izolována z buněk šetrně, pak je představována celými chromozomy, tedy mimořádně dlouhými molekulami. Pro mnoho molekulárně-genetických technik je proto nejdříve nutné takové molekuly rozštípat na menší úlomky, se kterými lze snáze pracovat.

Mechanicky lze molekulu DNA „polámat“ velmi účinně ultrazvukem, který rozkmitá dlouhé molekuly. Ty se pak v místech největšího mechanického namáhání v průběhu kmitání lámou. Místa zlomu jsou ovšem zcela nezávislá na konkrétní sekvenci DNA, jsou náhodná, takže v každém experimentu i v každé molekule dochází k lámání v jiných místech.

Nukleové kyseliny lze biochemicky štípat pomocí enzymů nukleáz. Revoluci v práci s nukleovými kyselinami přinesl objev tzv. restrikčních endonukleáz II. typu, které rozpoznávají v molekule DNA krátké specifické sekvence a v místě výskytu této sekvence molekulu štěpí. Revolučnost štěpení restrikčními endonukleázami není jen v tom, že jsme schopni štěpit reprodukovatelně, tj. ve stejném místě (pokud použijeme stejnou endonukleázu), ale také v tom, že rozštěpené molekuly můžeme znovu spojit.“ [1, str.18]

Většinou štěpíme větší množství molekul genomové DNA, takže ve výsledné směsi je každý specifický fragment zastoupen v mnoha kopiích, odpovídajících počtu vstupních molekul DNA. Takovou směs nemá smysl znovu spojovat, neboť bychom dostali mnoho náhodných spojení mezi různými fragmenty v různě přeházeném pořadí. Aby mělo štípání molekul DNA smysl, musíme mít možnost ve směsi rozpoznat a izolovat určitý fragment, obsahující například gen nebo určitou sekvenci, která nás zajímá.

Klonování DNA

„Klonováním DNA se myslí množení určitého úseku DNA. V molekulární genetice se ke klonování DNA s výhodou využívají enzymatické systémy živých buněk, které jsou schopny konkrétní DNA kopírovat a klonovat prakticky bezchybně a s nízkými náklady. Prakticky technika klonování znamená, že je třeba přimět určité buňky, aby kopírovaly úsek cizí DNA, který jsme předtím připravili nějakou jinou molekulárně-genetickou metodou a který po namnožení (klonování) v živých buňkách zase získáme zpět.“ [1, str.18]

Sekvenování DNA

„Termín sekvenování DNA znamená určení sekvence nukleotidů (resp. bází) v jednom z řetězců DNA. Důvodů pro zjišťování sekvence je mnoho. Pokud má například nějaká choroba genetický základ, pak je její konkrétní příčinou změna zdravé alely genu v chorobnou alelu. Podkladem této změny je právě konkrétní změna sekvence DNA. Podaří-li se takovou změnu po srovnání sekvencí zdravých a nemocných jedinců identifikovat, může to sloužit k vývoji diagnostického postupu, který bude tuto změnu namísto sekvenování identifikovat některou jednodušší a méně nákladnou molekulárně-genetickou metodou. Znalost změny sekvence, která je příčinou choroby, také otevírá cestu k budoucí cílené léčbě genovou terapií (náhradou „chorobné“ alely „zdravou“ alelou).“ [1, str.19]

Známe dvě metody – chemická Maxam-Gilbertova a biochemická Sangerova [1]. Druhá z nich dnes jednoznačně převládá. Vyžaduje jednovláknovou DNA, která slouží jako vzor (templát) pro syntézu komplementárního řetězce. Syntézu zprostředkuje enzymem polymeráza a reakce je upravena takovým způsobem, že v určitých místech dochází v závislosti na sekvenci templátového vlákna k ukončení (terminaci) syntézy. Reakce funguje tak, že k ukončení reakce dojde v okamžiku, kdy se v templátovém vlákně přečte určitý nukleotid. Délka nedokončeného úseku, zjištěná elektroforézou v gelu, pak odpovídá vzdálenosti od začátku vlákna, ve které se vyskytl určitý nukleotid.

Na základě tohoto ukončení se tedy identifikuje jedna ze čtyř bází. Pokud běží těchto reakcí v jeden okamžik paralelně více na několika dalších vláknech, je možné určit mnoho dalších bází, které se pak podle této délky komplementárního řetězce (který polymeráza stačila nasyntetizovat, než se zastavila) přesně zařadí pomocí počítačového programu do rekonstruované sekvence DNA.

3 Elektroforéza nukleových kyselin

Nyní si představíme elektroforézu nukleových kyselin. „*Elektroforéza patří v molekulární biologii k nejpoužívanějším separačním technikám při purifikaci, izolaci a analýze nukleových kyselin a bílkovin. Principem elektroforetické separace je pohyb nabitých molekul v elektrickém poli. Hlavním nositelem náboje nukleových kyselin jsou negativně nabité fosfátové skupiny, a proto se nukleové kyseliny v elektrickém poli pohybují k opačně nabité elektrodě – anodě.*“ [2, str.13]

3.1 Gelová elektroforéza

„*Z praktických důvodů se elektroforéza neprovádí přímo v roztoku, ale ve vhodném nosiči. Tím bývá obvykle gel. Elektroforetické gely používané pro separaci nukleových kyselin jsou nejčastěji tvořeny polyakrylamidem nebo agarózou, které vytvářejí složitou síťovou strukturu polymerních molekul s póry, jejichž velikost lze ovlivnit složením roztoku a koncentrací polymeru. Agarózové gely jsou vhodné pro separaci molekul nukleových kyselin o velikosti od 100 bp až po zhruba 50 kb, polyakrylamidové gely se používají pro separaci menších molekul (10 až 1000 bp). Podle polohy gelu v elektroforetické aparatuře rozlišujeme horizontální a vertikální gelovou elektroforézu, které mají deskové uspořádání,a dále kapilární elektroforézu, u níž je gel uvnitř kapiláry.*“ [2, str.13]

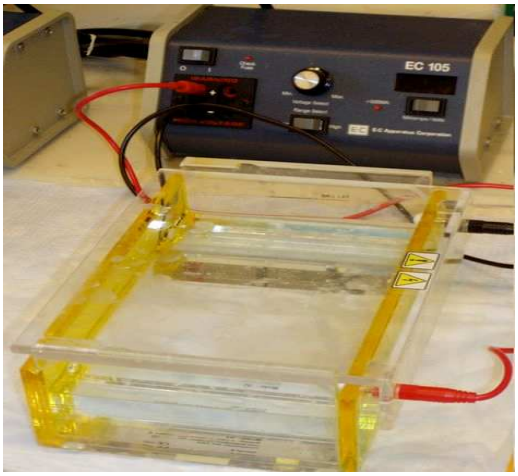
Rychlost pohybu molekul DNA v gelu označovaná jako elektroforetická pohyblivost [2] je nepřímo úměrná logaritmu jejich velikosti. Při posuzování elektroforetické pohyblivosti dané molekuly DNA není třeba brát v úvahu velikost náboje, protože nukleové kyseliny mají náboj rovnoměrně rozložen a jeho velikost je na jednotku délky molekuly stejná. Velikost molekuly DNA nebo jejího fragmentu o neznámé velikosti lze proto stanovit srovnáním jejich elektroforetické pohyblivosti s elektroforetickou pohyblivostí molekul nebo fragmentů DNA o známé velikosti, které se označují jako standardy velikosti nebo hmotnostní standardy. Těmi bývají většinou restrikční fragmenty plazmidových molekul nebo geonomu bakteriofágů, jejichž přesná velikost byla stanovena sekvencováním DNA.

Po dokončení elektroforézy je třeba identifikovat polohy separovaných molekul, které nejsou pouhým okem viditelné. Molekuly DNA lze snadno zviditelnit obarvením vhodným barvivem. Nejčastěji je používán etidiumbromid, který se vmezeřuje mezi sousední páry bází v DNA a vytváří s ní komplex, který po osvětlení ultrafialovým světlem červeně fluoreskuje. Molekuly DNA o stejné velikosti jsou pak na gelu patrné jako proužky, jejichž intenzita je úměrná koncentraci DNA. Etidiumbromid lze použít rovněž pro barvení RNA, intenzita jejího zbarvení je však nižší. Namísto etidiumbromidu může být pro barvení nukleových kyselin použita také skupina fluorescenčních kyaninových barviv s komerčním označením SYBR. Pro detekci molekul DNA separovaných v polyakrylamidových gelech se rovněž používá barvení stříbrem. Pokud jsou molekuly nukleových

kyselin raioaktivně označeny, znázorní se polohy proužků na gelu autoradiograficky. K detekci poloh fragmentů DNA lze využít rovněž hybridizace se značenou sondou.

„*Gelová elektroforéza může být využita rovněž pro separaci a studium molekul DNA nacházejících se v různých molekulárních typech (konformacích). Lze tak odlišit kovalentně uzavřené kružnicové molekuly DNA od molekul lineárních nebo od otevřených kružnic. Nadšroubovicová (superhelikální) forma, otevřená kruhová forma a lineární forma DNA téže molekulové hmotnosti se pohybuje agarozovým gelem různou rychlostí. Relativní pohyblivost těchto tří forem závisí na podmínkách elektroforézy.*“ [2, str.14]

Gelová elektroforéza, probíhající v aparátu k tomu určeném (obr. 3.1), je rovněž vhodným prostředkem pro studium interakcí mezi nukleovými kyselinami a proteiny, zejména těmi, které se vážou na specifické sekvence DNA (transkripční faktory, represory apod.). Metoda se označuje jako gelová zpomalovací analýza (retardační analýza, EMSA). Vychází z poznatku, že elektroforetická pohyblivost DNA se po vazbě proteinů snižuje.



Obr. 3.1 Aparát pro gelovou elektroforézu

3.2 Elektroforéza nukleových kyselin pro stanovení jejich sekvence

„*Jednořetězcové oligonukleotidy o délce několika desítek až stovek bází, které jsou výsledkem reakcí používaných při sekvencování DNA, lze účinně separovat v denaturačních polyakrylamidových gelech. Podmínky elektroforézy musí být takové, aby pohyblivost závisela pouze na délce jednovláknových fragmentů DNA, tedy je třeba zabránit vytváření lokálních dvojřetězcových struktur, jejichž přítomnost by mohla ovlivnit pohyblivost fragmentů a znemožnit přesné stanovení jejich velikosti. Proto se používají gely s vysokou koncentrací denaturujících látek (např. močoviny) a elektroforéza se provádí při teplotě 50-60 °C. Kombinace těchto dvou faktorů znemožňuje renaturaci.*“ [2, str.16]

4 Enzymy používané k úpravám nukleových kyselin

Manipulaci s purifikovanými nukleovými kyselinami podstatně usnadňuje existence enzymů, které umožní specifický zásah do jejich struktury.

Rozdělení enzymů podle substrátové specifity

Většina metod molekulární biologie je závislá na využití enzymů, jejichž substrátem jsou nukleové kyseliny. Předností enzymových reakcí je jejich přísná specifičnost a možnost pracovat s velmi malým množstvím materiálu. Enzymy používané při analýze a úpravách nukleových kyselin lze klasifikovat podle několika kritérií. Podle substrátové specifity se tyto enzymy dělí na dvě základní skupiny:

- enzymy, jejichž substrátem je DNA
- enzymy, jejichž substrátem je RNA

Substrátová specifita je obvykle absolutní, i když rozlišení mezi oběma druhy nukleových kyselin je podmíněno pouze přítomností nebo absencí 2‘-OH skupiny ribózy.

Rozdělení enzymů podle typu reakcí

Podle typu reakcí, které katalyzují, lze enzymy obou výše uvedených skupin dále dělit na:

- enzymy syntetizující nukleové kyseliny (polymerázy)
- enzymy modifikující nukleové kyseliny (fosfáty, metylázy, kinázy)
- enzymy spojující nukleotidové řetězce (ligázy)
- enzymy odbourávající nukleové kyseliny (nukleázy)

Podle substrátové specifity se enzymy jednotlivých skupin dále dělí, např. na DNA-polymerázy, RNA-polymerázy atd., což už dále nebude v této práci rozebíráno.

5 Amplifikace nukleových kyselin

POLYMERÁZOVÁ ŘETĚZOVÁ REAKCE (PCR)

Zavedení polymerázové řetězové reakce [2] v roce 1985 Kary B. Mullisem znamenalo pro molekulární biologii stejný přínos jako objev restrikčních endonukleáz nebo zavedení sekvencování DNA. Výhodou PCR je zejména to, že umožňuje získat požadovanou a specifickou sekvenci genomové DNA bez jejího předchozího klonování ve vektorech. Princip PCR je založen na replikaci nukleových kyselin, která je základním molekulárním procesem všech živých organismů. Podstatou PCR je cyklicky se opakující enzymová syntéza nových řetězců vybraných úseků dvojřetězcové DNA ve směru 5‘ -> 3‘ prostřednictvím DNA-polymerázy. Studovaný úsek nukleotidové sekvence je vymezen připojením dvou primerů, které se vážou na protilehlé řetězce DNA tak, že jejich 3‘-konce směřují proti sobě. Po přidání DNA-polymerázy a nukleotidů pak probíhá syntéza nových vláken na obou matricových řetězcích protisměrně.

K syntéze DNA se používají termostabilní polymerázy, což umožňuje, aby syntéza DNA probíhala opakovaně formou cyklů. PCR je proces, při němž se v závislosti na teplotě reakční směsi pravidelně střídají tři kroky, během nichž probíhají tři odlišné děje s odlišnými nároky na teplotu:

- denaturace dvojřetězcových molekul DNA (94 °C)
- připojení primerů k odděleným řetězcům DNA (30-65 °C)
- syntéza nových řetězců DNA prostřednictvím DNA-polymerázy (65-75 °C)

To nám k amplifikaci nukleových kyselin bude stačit.

6 Shluková a gradientová analýza

Tato kapitola čerpá z [5, 6, 7].

Shluk (*cluster*)

„*Je to skupina objektů, které uvnitř nějaké větší skupiny nemají ani nahodilý ani rovnoměrný výskyt a jejich vzájemná vzdálenost, resp. nepodobnost je menší než vzdálenost, resp. nepodobnost s objekty, které patří do jiných shluků.*“ [5, str.1]

Shlukování

Nemáme žádné informace o existenci skupin a chceme klasifikovat všechny sledované objekty (chceme vytvořit shluky). Nejsou předem známé vlastnosti tříd (učení bez učitele).

Shluková analýza

„*Postup formulovaný jako procedura, pomocí níž objektivně seskupujeme jedince do skupin na základě jejich podobnosti a odlišnosti (zkráceně R. C. Tryon, 1939).*“ [6, str.2]

Použitelné k získání informace o rozdělení dat, pro další zpracování - například:

- k rozpoznávání obrazů (vzorů)
- k datové analýze
- k výzkumu trhu

Obecné poznámky ke shlukovacím metodám

„*Pokud data nemají zcela jednoznačnou a zřetelnou strukturu (jedná se víceméně o náhodně rozptýlené objekty), je pravděpodobné, že použití různých shlukovacích technik přinese odlišné výsledky.*

Pokud různé shlukovací techniky přinášejí z téhož souboru dat shodné, resp. podobné výsledky, je to do jisté míry potvrzení struktury obsažené v datech (ačkoliv shlukovací metody patří k postupům produkujícím hypotézy a nejsou určeny k jejich testování).

Mnohé shlukovací techniky jsou citlivé na přítomnost odlehlých objektů (outliers, výrazně atypických případů). Před samotnou shlukovou analýzou je proto vhodné použít některou z metod na jejich detekci. Výrazně odlehlé objekty se zpravidla z dalších analýz vylučují.

Shlukové analýzy obecně nejsou vhodné pro data, která popisují klinální variabilitu znaků (cline = variabilita znaku závislá na gradientu prostředí).“ [5, str.19]

Typy dat při shlukové analýze

Předpokládejme do datové matice (relační tabulky) uspořádaných

- n objektů
- majících p atributů

(tzv. NxP matice) D =

$$\begin{bmatrix} x_{11} & \dots & x_{1p} \\ : & & : \\ x_{n1} & \dots & x_{np} \end{bmatrix}$$

objekt t_1
:
objekt t_n

Lze vyjádřit i maticí rozdílnosti (vzdálenosti) objektů A =

$$\begin{bmatrix} 0 & d(2,1) & d(1,n) \\ d(2,1) & 0 & \\ : & & : \\ d(n,1) & \dots & 0 \end{bmatrix}$$

kde (dissimilarity) $d(i, j)$ je mírou vzdálenosti (nepodobnosti) objektů s indexy i a j.

Volba míry (ne)podobnosti závisí na typu proměnných (znaků):

Kvalitativní znaky

- Nominální (profese, typ školy)
- Ordinální (hodnocení výrobku)
- Binární (0 / 1) :

Kontingentní (kontingenční, asociační) tabulka pro 2 objekty (vzorky) (tab. 6.1) [6]:

		ve vzorku A	
		přítomen	nepřítomen
ve vzorku B	přítomen	*a*	*b*
	nepřítomen	*c*	*d*

Tab. 6.1 Kontingentní tabulka pro 2 objekty binárních znaků

a počet proměnných rovných 1 u obou objektů

d počet proměnných rovných 0 u obou objektů

c počet proměnných rovných 1 pro object A, ale 0 pro B

d počet proměnných rovných 0 pro object A, ale 1 pro B

Užívají se zejména tyto koeficienty:

Jaccardův koeficient: $J = a / (a + b + c)$

Sörensenův koeficient: $S = 2a / (2a + b + c)$

Euklidovský koeficient: $E = \sqrt{b + c}$

Kvantitativní znaky

- Intervalové (hodnota ve stupních Celsia)
- Poměrové (počet členů domácnosti)

Užívá se euklidovská vzdálenost, manhattanská metrika, tětivová vzdálenost.

Taxonomie shlukovacích metod

Nehierarchické metody (partitioning methods)

Rozkládají data.

- *k*-means - každý shluk je reprezentován střední hodnotou objektů ve shluku. Časová náročnost je $O(t\,kn)$, kde t je počet iterací
- *k*-medoids - každý shluk je reprezentován jedním z objektů, umístěným blízko středu shluku
- neuronové sítě, Kohonenovy sítě (mapy)
- metody založené na hustotě (shluk narůstá, pokud počet objektů v sousedství překračuje zadanou mez)
- metody založené na mřížkách (prostor objektů rozparcelují mřížkou a shluky hledají na buňkách mřížky)

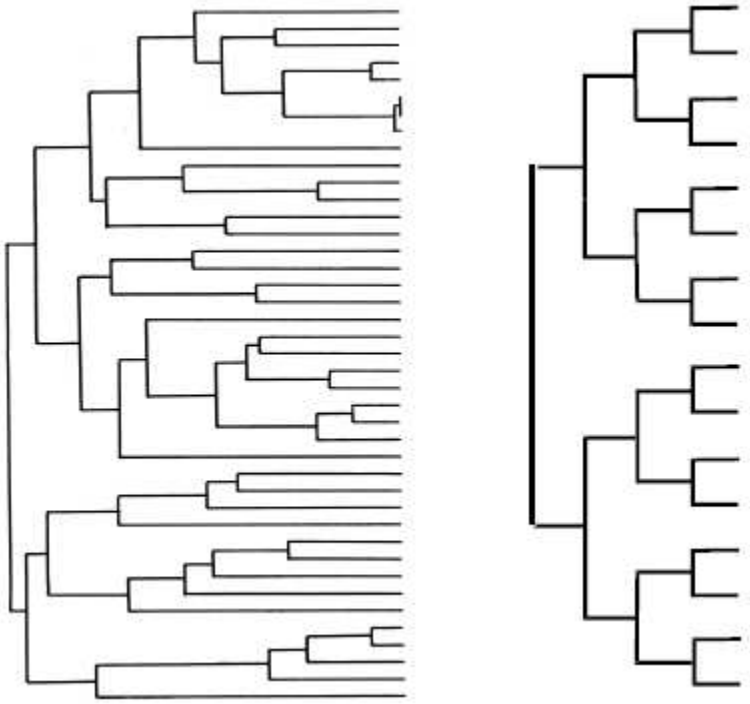
Hierarchické metody

Vytváří stromovou strukturu – dendrogram (obr. 6.1)

Prostorová náročnost je $O(n^2)$ – matice rozdílnosti

Časová náročnost je $O(kn^2)$ – pro každou úroveň dendrogramu jedna iterace

- aglomerativní (bottom-up)
  na počátku každý objekt je shlukem, postupně se shluky sdružují, dokud není splněna ukončovací podmínka
- divisivní (top-down)
  na začátku jsou všechny objekty v jednom shluku a postupně jsou shluky štěpeny na menší



Obr. 6.1 Příklad aglomerativního (vlevo) a divizního dendrogramu (vpravo)

Aglomerativní algoritmy:

„*Vstup:*

D = { t 1, t 2 , …, t n } množina objektů

A matice vzdálenosti (nepodobnosti), předp. hodnoty 0, 1, 2, …

Výstup:

DE dendrogram tvaru <d, k, K>, kde d je prahová vzdálenost,

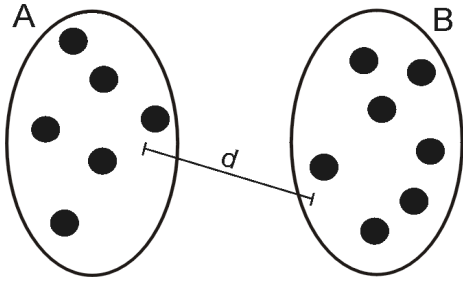
k je počet shluků, K je množina shluků

Postup:

```
d = 0; na začátku je každý objekt shlukem, vzdálenost objektu je 0
k = n;
K = {{ t 1 }, { t 2 }, …, { t n }};
DE = {<d, k, K>};
repeat
  oldk = k;
  d = d + 1;
  A d = matice s prvky prahové vzdálenosti d;
  <k, K> = NewClusters(A d, D );
  if oldk ≠ k then
      DE = DE ∪ <d, k, K>;  přidání nových shluků k dendrogramu
until k = 1;
```

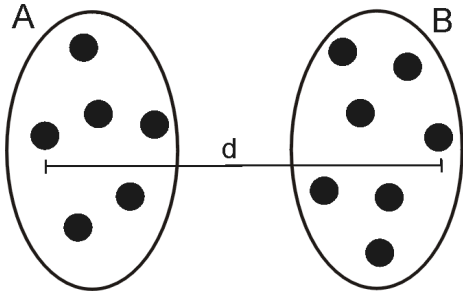
Různé aglomerativní algoritmy se liší procedurou NewCluster. Liší se i způsobem určování vzdálenosti shluků: single link, complete link, average link, apod. (viz níže). Pro rozsáhlé databáze se příliš nehodí - velká náročnost paměťová i časová (složitý výpočet nových shluků v iteraci). Neschopnost pracovat inkrementálně. Při změně prvku nutno přepočítat celé.“ [7, str.7]

Metoda nejbližšího souseda (jednospojná metoda, metoda jediné vazby, *single linkage, the nearest neighbor method,* obr. 6.2). Vzdálenost mezi dvěma shluky je minimem ze všech vzdáleností mezi objekty.



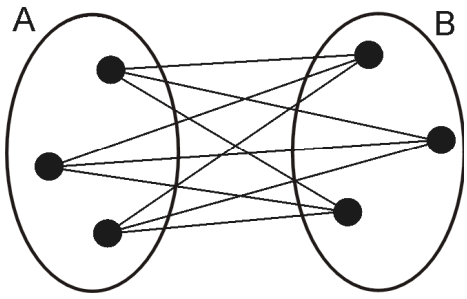
Obr. 6.2 Metoda nejbližšího souseda [5]

Metoda nejvzdálenějšího souseda (všespojná metoda, metoda úplné vazby, *complete linkage*, *the furthest neighbor method,* obr. 6.3). Vzdálenost mezi dvěma shluky je maximem ze všech vzdáleností.



Obr. 6.3 Metoda nejvzdálenějšího souseda [5]

Metoda průměrné vzdálenosti (středospojná metoda, metoda průměrné vazby, *average linkage, UPGMA – unweighted pair-group method using arithmetic averages,* obr. 6.4). Vzdálenost mezi dvěma shluky je průměrem ze všech vzdáleností mezi jejich objekty.



Obr. 6.4 Metoda průměrné vzdálenosti [5]

Některé z dalších metod jsou:

Centroidní metoda (Gowerova metoda, *centroid method, UPGMC – unweighted pair-group method using centroids*)

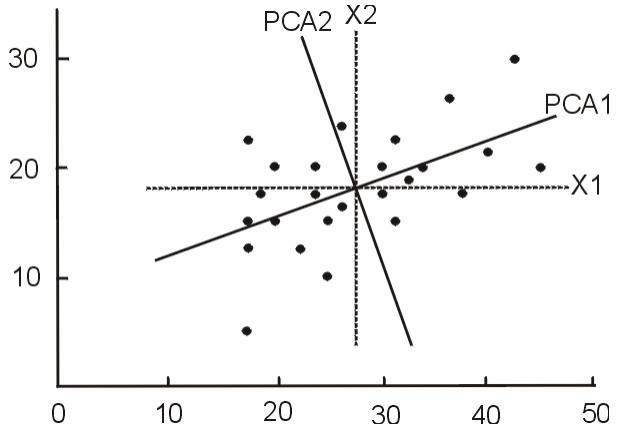
Mediánová metoda (*median method, WPGMC – weighted pair-group method using centroids, weighted centroid clustering*)

Ordinační metody

„Objekty charakterizované p znaky je možné si představit jako body v p rozměrném prostoru, kde každý z rozměrů představuje hodnoty jednoho znaku. Pokud pracujeme pouze se dvěma nebo třemi znaky, je možné bez problémů sledovat na dvoj- případně trojrozměrném grafu vztahy mezi objekty, jejich vzdálenosti a seskupení. Větší počet znaků vyžaduje nutnost redukce jejich počtu s co nejmenší ztrátou informace.“ [5, str.24]

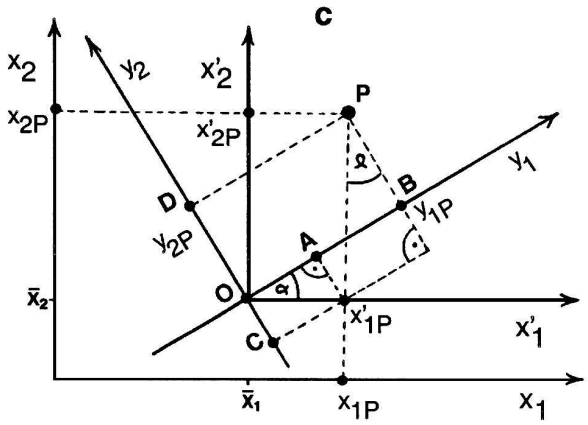
Analýza hlavních komponent (PCA – *principal component(s) analysis*)

„Nahrazuje původní soubor pozorovaných znaků souborem nových (hypotetických), vzájemně nekorelovaných znaků tak, že první nová osa (první hlavní komponenta, PC1, první nový znak) je vedena ve směru největší variability mezi objekty, druhá osa (druhá hlavní komponenta, PC2, druhý nový znak) je vedena ve směru největší variability, který je kolmý na směr první komponenty, atd. (viz obr. 6.5)“ [5, str.26]



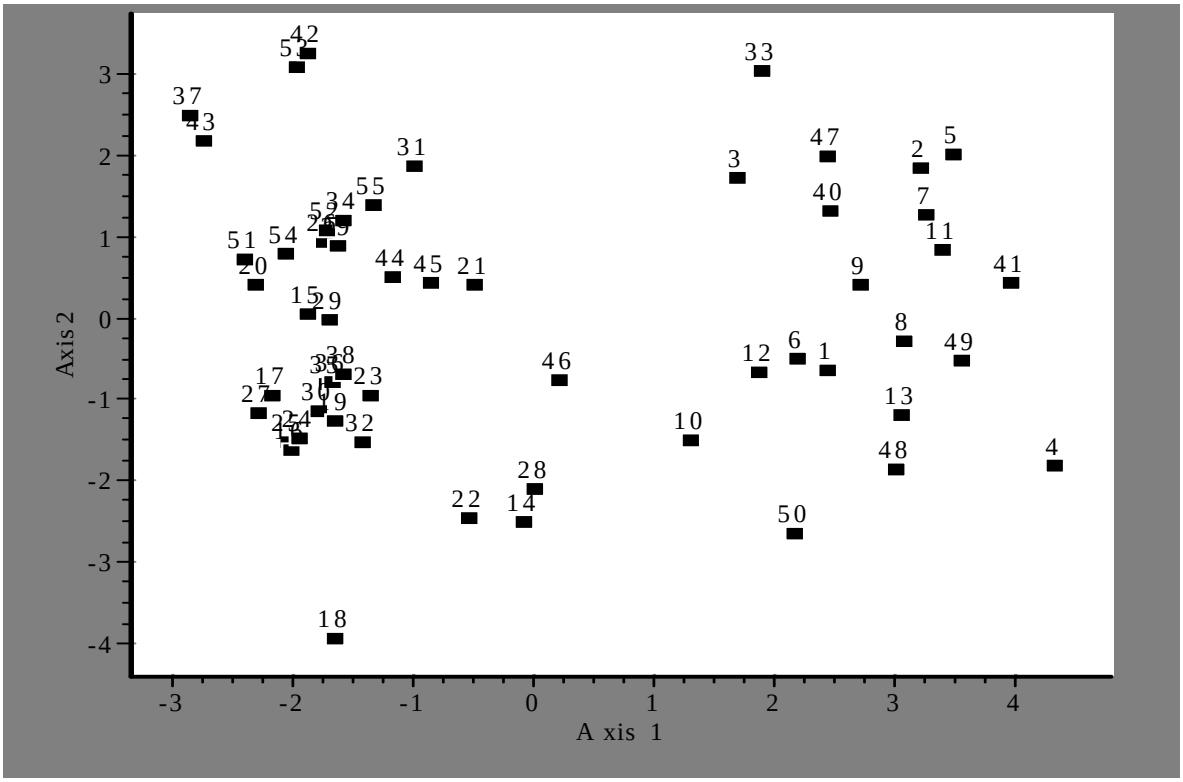
Obr. 6.5 Princip PCA [5]

Je založena na vlastní analýze (eigenanalysis) symetrických matic (korelační, kovarianční matice). Cílem PCA je určení úhlů mezi původními a novými osami souřadnicové soustavy, souřadnice objektů v novém systému souřadnic (obr. 6.6). Jde tedy o nalezení vlastních vektorů kovarianční nebo korelační matice.



Obr. 6.6 Určení nového systému souřadnic [5]

„Počet objektů při PCA musí být alespoň o jeden větší než je počet analyzovaných znaků. Obvykle se však doporučuje, aby se počet objektů blížil druhé mocnině počtu znaků (souvisí s počtem stupňů volnosti). V případě, že n ≤ p (kde n je počet objektů a p počet znaků), výsledná matice (korelační nebo kovarianční) řádu p má jen n – 1 nezávislých řádků nebo sloupců. V takovém případě příslušná matice má p – (n – 1) nulových vlastních čísel (na umístnění n objektů podle jejich vzájemných vzdáleností je zapotřebí jen n – 1 rozměrů). Pokud jsou např. dvě základní skupiny (oddělené podél první osy) uvnitř členěné komplikovanějším způsobem, bývají druhá, třetí a další osy jistým kompromisem mezi strukturou v obou základních skupinách, je tedy vhodné každou skupinu (oddělenou podle první osy) analyzovat v dalších krocích samostatně. Ačkoliv byla technika PCA původně navržena pro kvantitativní znaky, může se použít také pro znaky binární a semikvantitativní. Binární data však ve zvýšené míře způsobují tzv. „podkovový efekt“ (horseshoe effect), kdy jsou objekty v ploše definované prvními dvěma komponentami uspořádané do tvaru podkovy. Toto zakřivení odstraňují tzv. detrendované techniky, v taxonomických aplikacích se však podobná „narovnání“ zpravidla nepoužívají.“ [5, str.32, str.33, str.34] Příklad výsledku této metody je možno vidět na obr. 6.7.



Obr. 6.7 Příklad výsledku PCA

Analýza hlavních koordinát (PCoA) (metrické mnohorozměrné škálování, PCoA – *principal coordinate(s) analysis, metric multidimensional scaling, classical scaling*)

„Jedná se o rozmístění souboru objektů v novém prostoru definovaném hlavními koordináty (novými osami). Vzájemné (euklidovské) vzdálenosti objektů odrážejí vztahy mezi původními objekty měřené libovolným koeficientem podobnosti nebo vzdálenosti. Znaky mohou být binární, vícestavové kvalitativní či smíšená data.“ [5, str.45]

Postup analýzy:

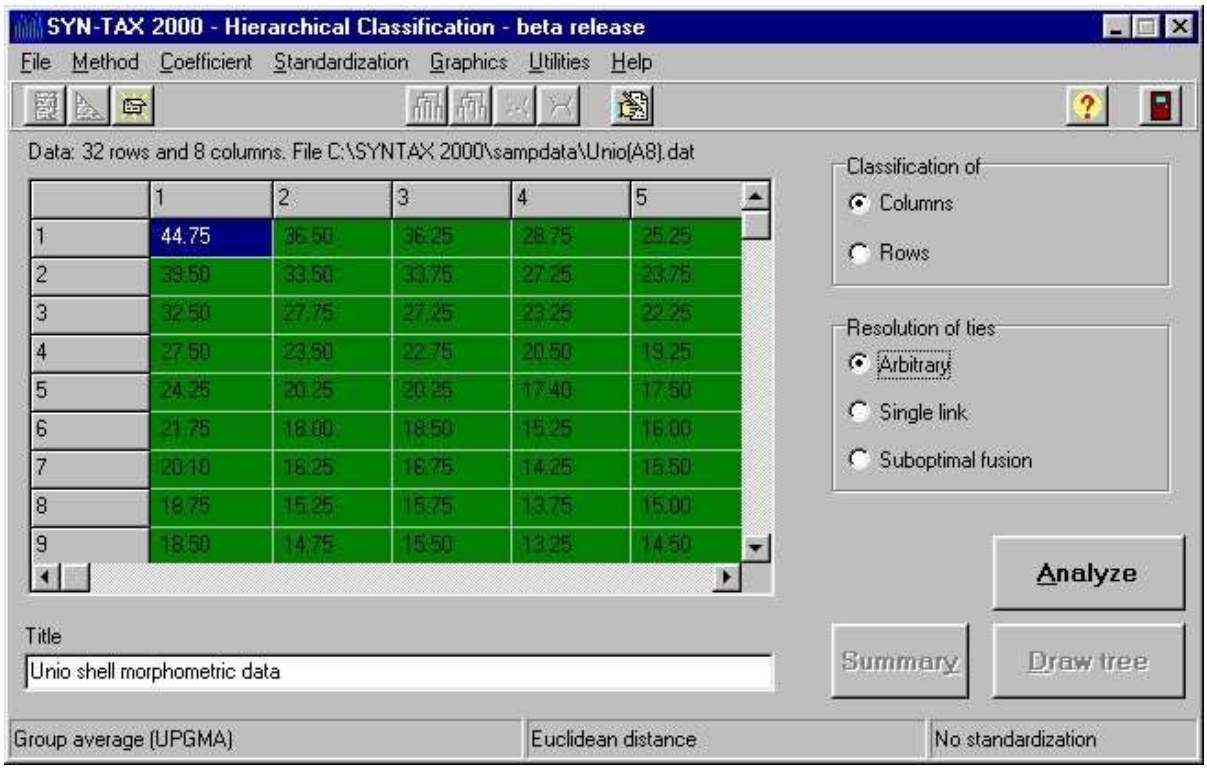
(1) Primární matice dat, sekundární matice vzdáleností, symetrická matice (ekvivalentní korelační nebo kovarianční matici používané v PCA).

(2) Výpočet vlastních čísel, vlastních vektorů a komponentních skóre.

Souřadnice v prostoru určeném hlavními koordinátami nejsou lineárně závislé na hodnotách původních znaků. Lze vhodně použít také tehdy, pokud počet znaků převyšuje počet objektů (např. u molekulárních dat).

7 Existující nástroje

Při zkoumaných metodách se běžně používá program Syntax 2000 [4]. Je to program pro analýzu dat v oblasti ekologie a taxonomie (třídění organismů podle původu). Pracuje v prostředí operačního systému Windows. První vydání programu se skládá ze tří modulů. Je to modul pro hierarchické metody (obr. 7.1), ordinální metody a nehierarchické metody. Blíže si uvedeme první dva jmenované.

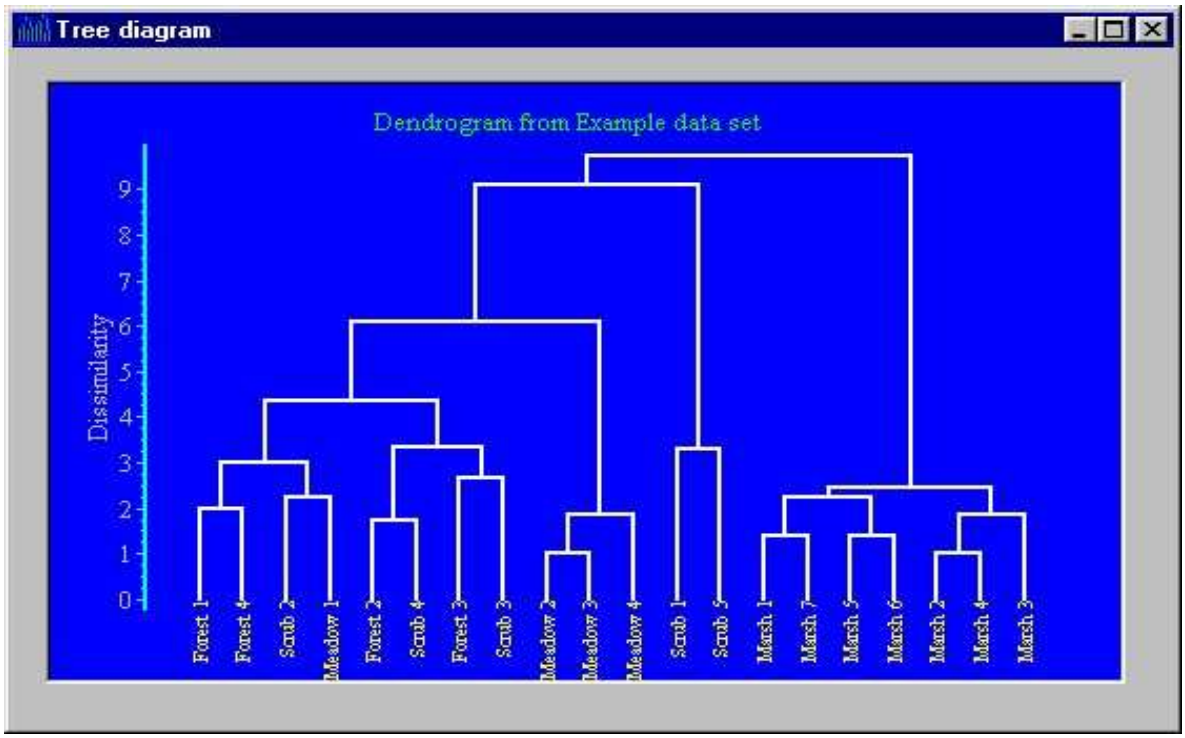


Obr. 7.1 Program Syntax 2000 – modul pro hierarchické metody [4]

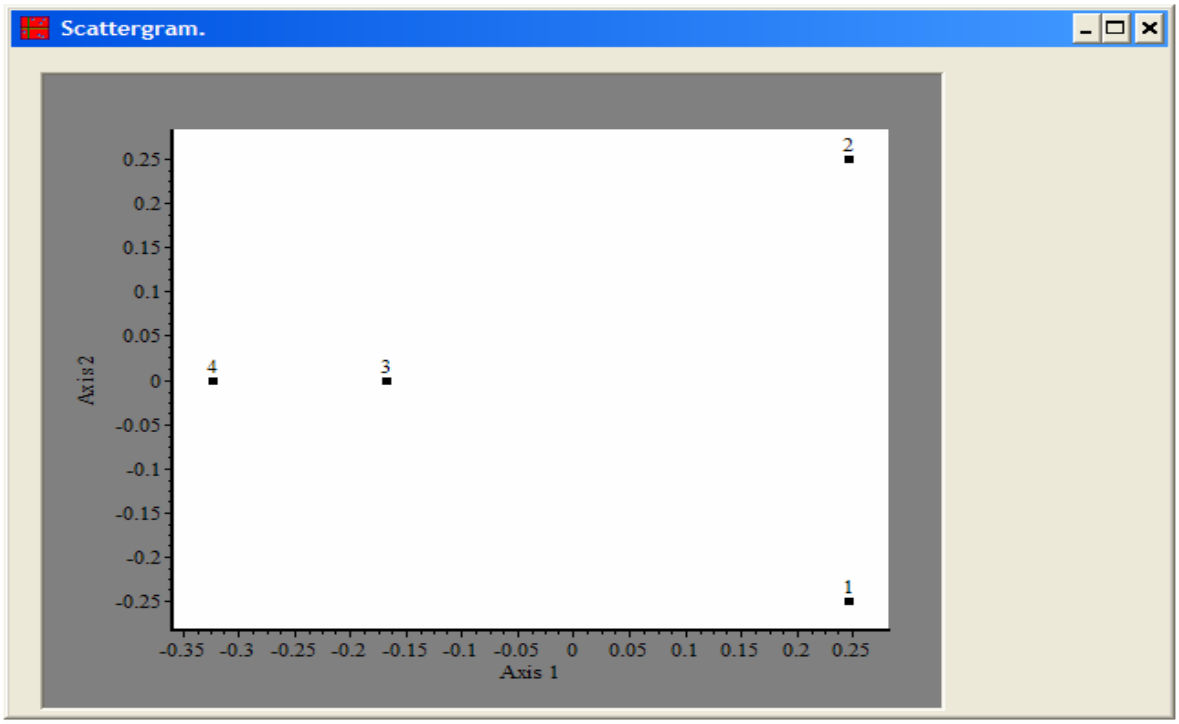
Moduly jsou si velmi podobné, jediný rozdíl spočívá v odlišné nabídce metod, dle kterých se matice podobnosti bude zpracovávat. Oba moduly tedy umí otevřít jak syrová data, tak již distanční matici. Po otevření syrových dat lze navíc zvolit koeficient podobnosti pro kvalitativní, kvantitativní či mixovaná data. Výstupem modulu pro hierarchické metody je dendrogram (obr. 7.2) a výstupem pro ordinální metody je scattergram (také nazývaný scatter plot, obr. 7.3).

Pro naše účely se zaměříme na případ, kde vstupní formát musí být soubor *.dat (program zde očekává zadaná data ve formátu: první řádek popis, druhý řádek obsahuje počet řádků a sloupců matice oddělený mezerou, následujícím řádkem počínaje pak samotná matice). Soubory *.xls se dají převést do požadovaného formátu pomocí podprogramu Syntax Converter (obr. 7.4). Nás nejvíce zajímají binární data. U nepřítomnosti vzorku v matici (prázdné místo) ovšem sám nepřiřadí 0, u přítomnosti vzorku (libovolné číslo) lze jeho hodnotu změnit na 1, ale pouze pomocí utility Data

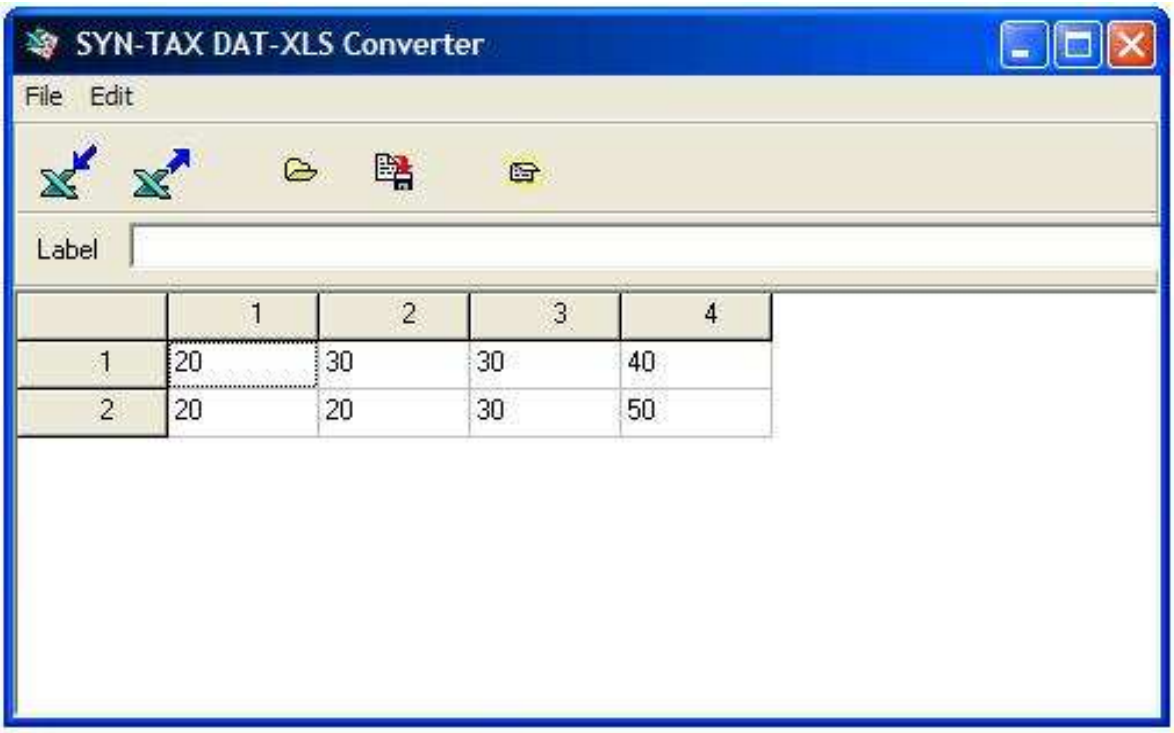
Standardization - Binarization v záložce Utilities. Tyto nedostatky a nepodpora jiných vstupních souborů bez předchozích převodů jsou zápory, které vedou uživatele k potřebě jiného programu. Může jít o jednodušší aplikaci, ale takovou, která umí dělat přesně to, co je potřeba, bez dalších zásahů automaticky.



Obr. 7.2 Příklad dendrogramu dvaceti vzorků programu Syntax 2000 [4]

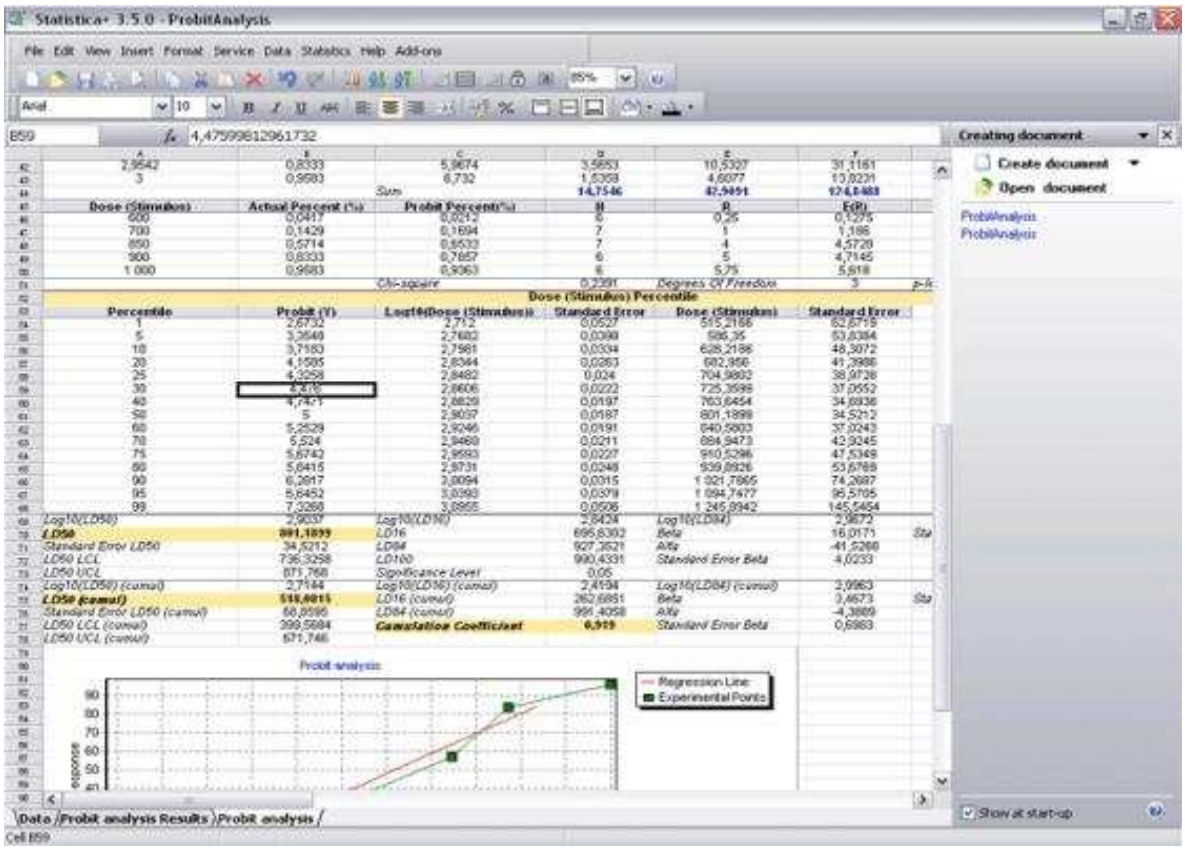


Obr. 7.3 Příklad scattergramu čtyř vzorků o dvou osách programu Syntax 2000 [4]



Obr. 7.4 Podprogram Syntax Converter [4]

Dalším programem je Statistica [8]. Podporuje mnoho odvětví včetně analýzy dat (obr. 7.5).



Obr. 7.5 Ukázkový příklad programu Statistica 3.5.0 [8]

8 Analýza a návrh aplikace

V této kapitole navrhneme desktopovou aplikaci, která bude odpovídat požadavkům zadavatele.

8.1 Neformální specifikace

Zadavatel požaduje vytvoření aplikace řešící výpočetní problémy z oblasti genového inženýrství. Jde především o metody fragmentační analýzy DNA. DNA vzorků se zpracuje (výše popsanými metodami) na fragmenty různé délky a různého počtu. Ty budou vkládány pomocí desktopové aplikace do centrální databáze (připojení do databáze přes login a heslo, tím je dáno zabezpečení), odkud si je budou moci uživatelé touto aplikací vybrat ke srovnání. Vznikne tím neupravená matice, kde se v jednotlivých řádcích nacházejí vzorky se svými fragmenty. Nyní je potřeba upravit matici tak, aby podobné fragmenty vzorků byly ve stejných sloupcích dle nějakého zvoleného rozsahu. Algoritmem z neupravené matice vznikne upravená matice. Dále se z upravené matice vytvoří matice přítomnosti a nepřítomnosti jednotlivých údajů na pozici v matici, a to tak, že je-li fragment přítomen, uloží se na jeho pozici 1, a je-li nepřítomen, uloží se na jeho pozici 0. Takto upravená data do binární podoby jsou vstupními údaji pro matici podobnosti. Zvolí se koeficient (viz kapitola číslo 6 – Shluková a gradientová analýza), dle něj se vypočítají podobnosti vzorků a vznikne matice podobnosti. Matice podobnosti je tedy cílem. Můžeme ji pak podrobit buď analýze dle shlukových metod, nebo analýze dle ordinačních metod a výsledek si nechat zobrazit na grafu rozptýlení.

Centrální databáze

Jde o databázi druhů rostlin. Každý druh byl zpracován dvěma enzymy (enzym1, enzym2 – jsou to zkratky, např.: EcoRI, omezený počet zkratek, nelze zadat jakoukoliv, mělo by být formou výběru). Dále tam hraje roli, jakými primery a jejich extenzemi byl vzorek zpracováván, přičemž nutné bude si zřejmě pamatovat pouze extenze. Tyto extenze jsou tedy dvě pro preselekci a dvě pro selekci. Druh byl rozkapán nějakou specifickou barvou (opět jako u enzymů – zkratky, např.: Joe, Fam, Ned.., omezený počet zkratek, mělo by být formou výběru) několikrát. Tyto pokusy jsou známy jako vzorky (zkratka vzorku bude obsahovat k jakému druhu se váže a kolikátý je to pokus u vzorku), které mají určitý počet fragmentů (přibližně 50 - 1000). Fragmenty mohou být délky 20 až 500 (běžně čísla jako 55,5; 43,28; 174,9 atd.).

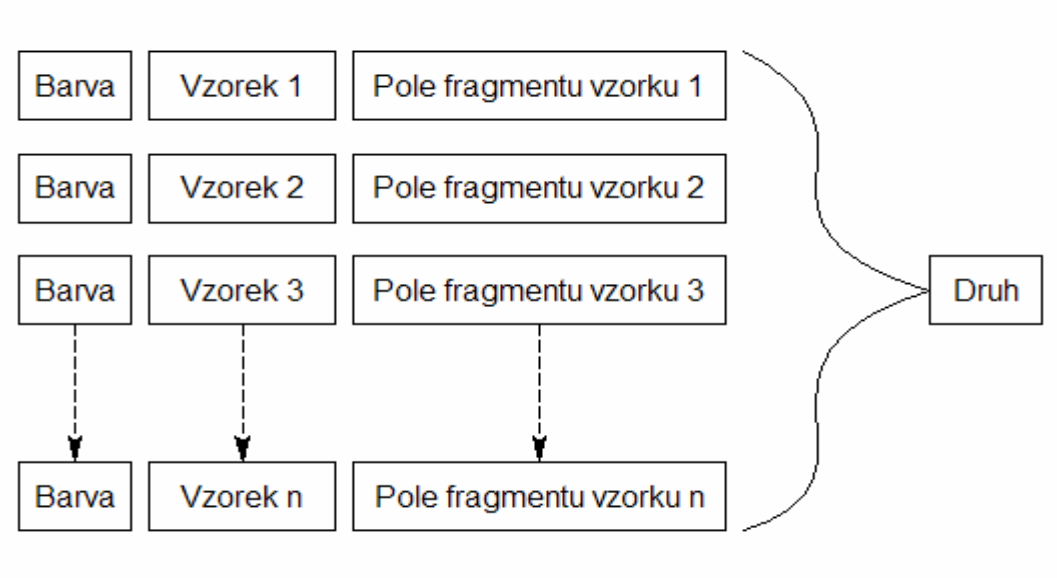
Zadání nového druhu:

- název, komentář, enzym1, enzym2, extenze1, extenze2, extenze1, extenze2

Zadání nového vzorku:

- druh (k jakému druhu vzorek patří)
- barva (jakou byl vzorek rozkapán)
- pole fragmentů (např.: 1000 čísel v rozsahu 20-500) obsahující v prvním sloupci označení vzorku

Struktura je tedy následující:



Vzorky 1-3 byly například zpracovány barvou Fam, další vzorky barvou Joe.

Klient při srovnávání dvou vzorků bude postupovat takto:

Nechá si vypsat druhy, které jsou v databázi s tím, jakými barvami byly jednotlivé druhy zpracovány.

Najde například:

A: Joe,Fam,Ned B: Joe,Fam

Srovnání druhu A a B:

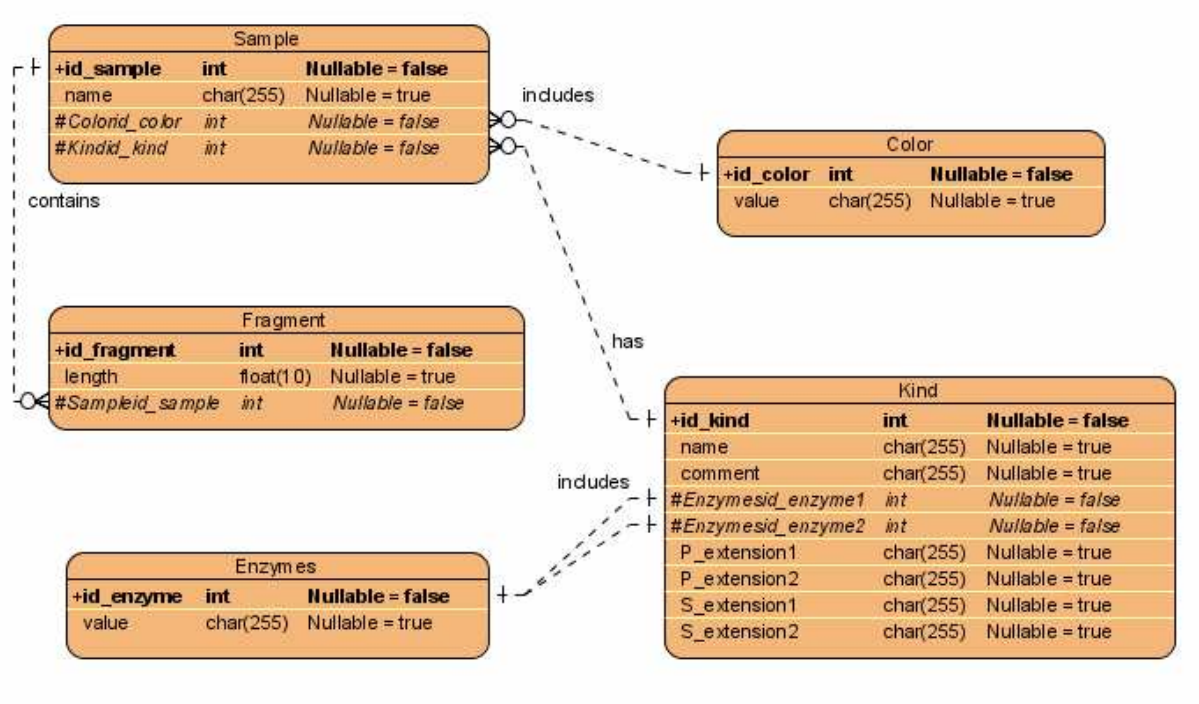
1. oba dva enzymy musí odpovídat (u A i B musí být například kombinace MseI-San3A)
2. preselekce: extenze1 i extenze2 musí odpovídat
3. selekce: extenze1 i extenze2 musí odpovídat
4. barva by měla odpovídat, ale nemusí

Uživatel si tedy výběrem zvolí, které barvy těchto dvou druhů bude srovnávat. Příklad zvolení:

A: Joe,Fam B: Joe,Fam

Výstupem této fáze je vypsání všech vzorků (a jejich fragmentů) z databáze, které mají příslušné barvy příslušných druhů.

8.2 ER diagram



8.3 Požadavky

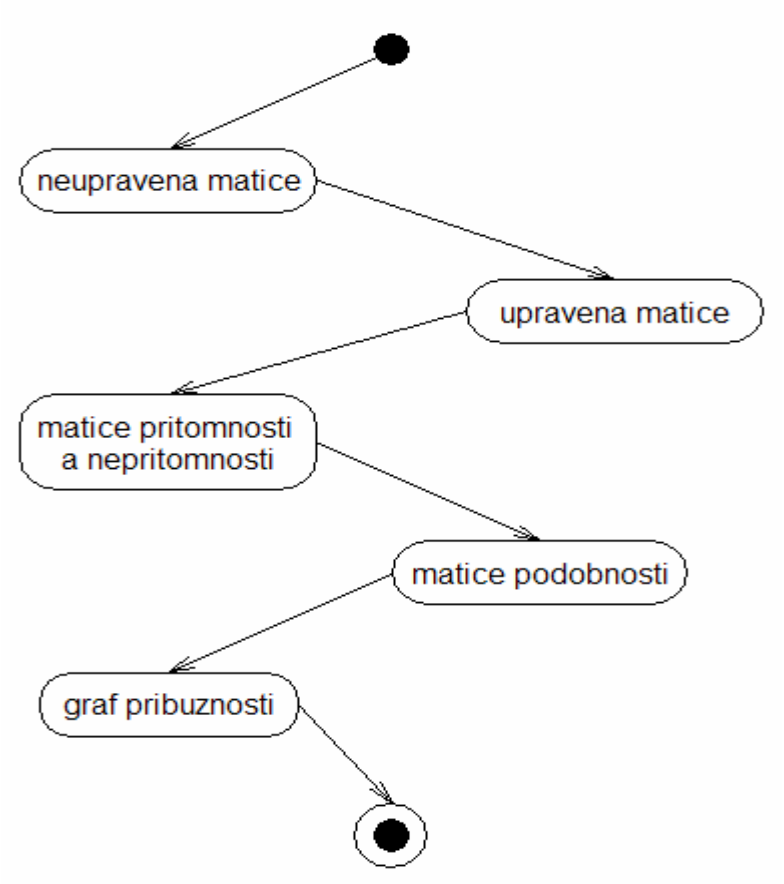
8.3.1 Funkční požadavky

- Aplikace má zobrazovat jednotlivé kroky úpravy matic od neupravené matice přes upravenou matici a matici přítomnosti a nepřítomnosti fragmentů až po matici podobnosti.
- Musí být umožněno zadávat přesnost, s jakou má být upravená matice vytvořena.
- Výběrem půjde volit koeficient, podle kterého se vypočítají podobnosti vzorků .
- Matice podobnosti musí být snadno složitelná do formátu, který bude možno vidět na diagramech příbuznosti.
- Vyžaduje se pouze jeden typ uživatele (jedna role), který bude obsluhovat aplikaci.

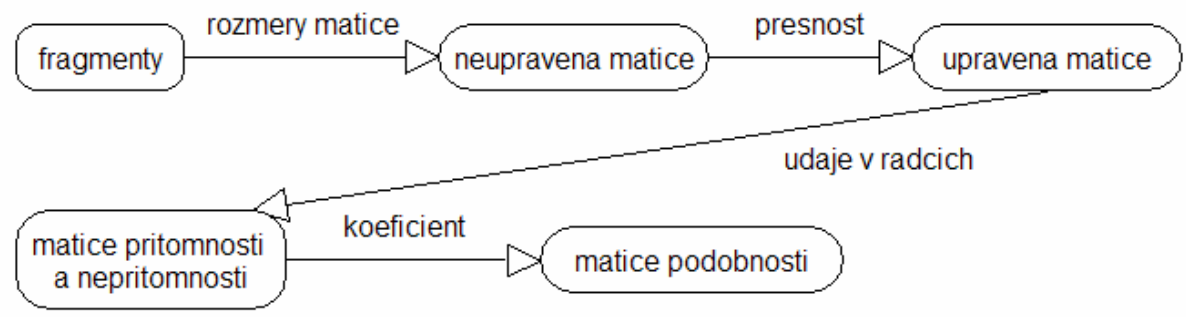
8.3.2 Nefunkční požadavky

- Aplikace musí běžet dostatečně rychle na běžném kancelářském počítači.
- Mělo by být zajištěno intuitivní a jednoduché ovládání (existuje možnost zaškolení).
- Aplikace by měla být snadno rozšiřitelná a odolná vůči chybám.
- Pro uživatele bude vytvořena dokumentace pro práci se systémem.
- Aplikace má být vytvořena cca do poloviny května roku 2008.

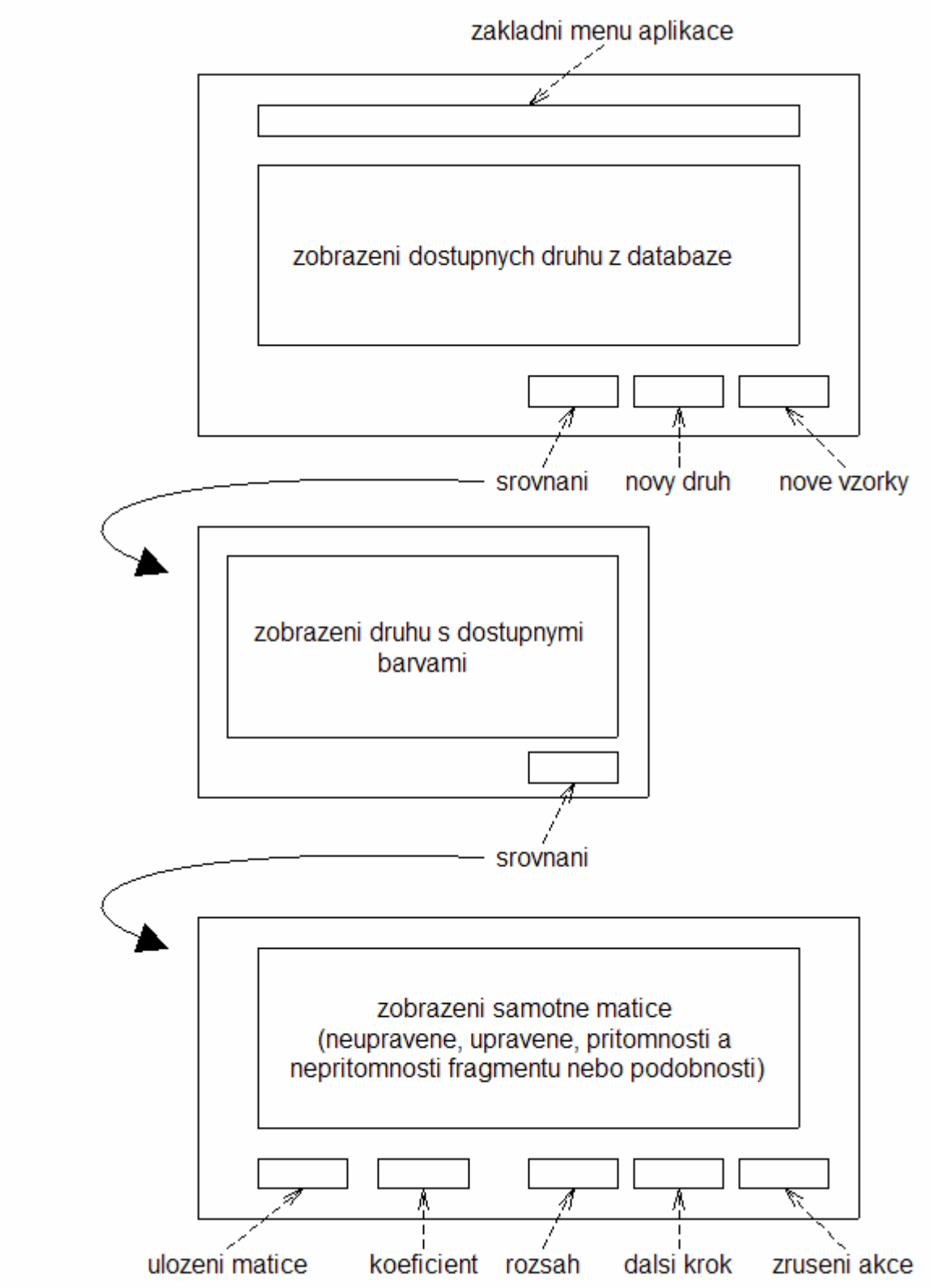
8.4 Stavový diagram celého procesu



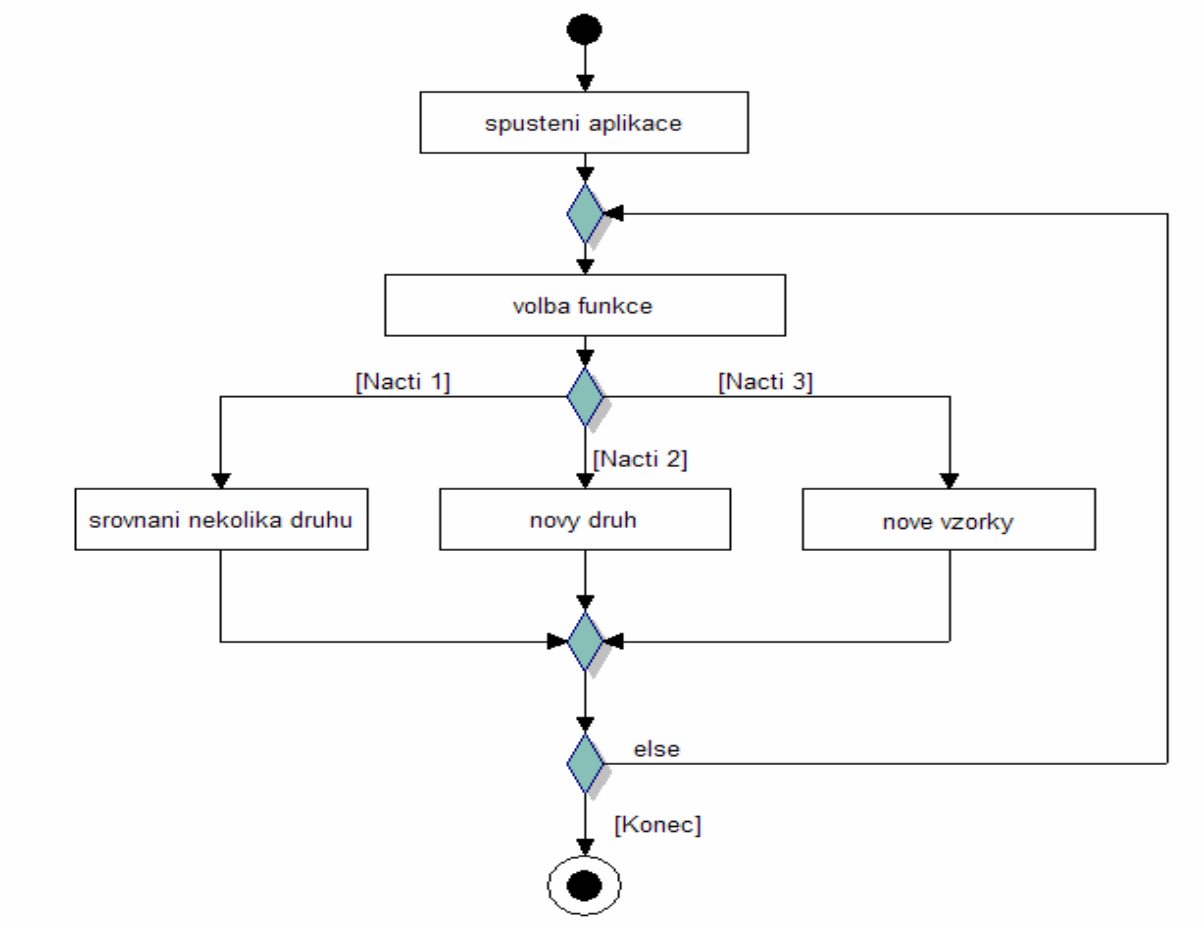
8.5 Diagram návaznosti jednotlivých kroků s volbou parametrů



8.6 Návrh prvků a jejich rozložení v oknech aplikace



8.7 Přehledový diagram interakce



8.8 Navržený algoritmus na úpravu matice

1. Projdi postupně všechny sloupce matice (sloupce mohou v průběhu algoritmu dynamicky vznikat, na začátku tedy nevíme, kolik sloupců bude po skončení algoritmu matice mít)
2. Pro každý sloupec najdi minimální číslo, které obsahuje
3. Projdi všechna čísla ve sloupci a porovnej je s minimem
4. Pokud je číslo v rozsahu s minimem (tolerance přesnosti), tak ho ve sloupci ponechej, jinak vlož nulovou hodnotu na pozici v matici (a tím se také hodnoty v inkriminovaném řádku posunou doprava)

Klíčové je dát si pozor na správné vyhledání minima ve sloupci, zvláště když jsou řádky nestejně dlouhé. Dynamickým přídáváním sloupců určitých řádků se nesmí porušit chod algoritmu. Bude tedy nutné najít takový datový typ, který bude pro naše účely vhodný.

9 Implementace

Programovací jazyk C#

- objektově orientovaný jazyk odvozený z jazyka C++, nicméně podporuje všechny konstrukce procedurálního jazyka
- byly eliminovány nebezpečné rysy jazyka C/C++ (ukazatelé, přímé adresování paměti, doplněna typová kontrola proměnných)
- byla doplněna automatická správa paměti (garbage collection)
- je postaven nad platformou .NET a může využívat její bohatou knihovnu funkcí

Platforma .NET

Jazyk C# je postavený na platformě .NET [9], což mu sice přináší mnoho výhod, na druhé straně potřebuje pro vývoj i spouštění přeložených programů mít tuto platformu nainstalovanou ještě přes MS Windows. Proto je třeba mít k dispozici tato balení platformy .NET :

- *MS NET Framework verze 1.1* - jako runtime na počítače, na kterých poběží vytvořené a přeložené programy v C#. Balíček má velikost ca 23MB a je přístupný ke stažení přes Službu Stažení softwaru společnosti Microsoft pod označením "Balíček k opětovné distribuci rozhraní Microsoft® .NET Framework verze 1.1".

-*NET Framework SDK* jako základ vývojového prostředí na počítači, na kterém budou programy vyvíjeny. Balíček má velikost ca 128MB a je přístupný ke stažení přes *Microsoft .NET Framework Developer Center.* I na tomto vývojovém počítači je třeba mít předem nainstalovaný již zmíněný - MS NET Framework verze 1.1.

SDK obsahuje pouze řádkové programy pro překlad, takže je dobré použít produktivnějšího nástroje. Využil jsem vývojové prostředí *Microsoft Visual Studio 2005* (které je pro studenty VUT zdarma k dispozici).

Práce s databází

ADO.NET

"*Je potřebné přistupovat k datům uloženým v nějakém datovém zdroji, kterým často bývá relační databázový systém, ale mohou to být i jiné. Cestou, jak k těmto datům přistupovat, je v prostředí .NET jeho součást, nebo jinými slovy komponenta, ADO.NET. Tato komponenta je následníkem technologie společnosti Microsoft s názvem ADO. Oproti této verzi, která byla určena pro práci z neřízeného kódu, byla velká část architektury přístupu k datům přepracována a nyní je její hlavní využití z kódu řízeného. Oproti staršímu ADO je již technologie ADO.NET mimo jiné připravena na použití v odpojených aplikacích (bude upřesněno níže) a také na nasazení ve vícevrstvých aplikacích, které jsou v dnešní době následníky aplikací s architekturou klient-server. Jednou*

z dalších důležitých vlastností komponenty ADO.NET je, že je možné standardizovaným způsobem přistupovat k datům v různých datových zdrojích. Můžeme tedy říci, že ADO.NET je soubor typů (tříd, rozhranní, výčtů…), které můžeme použít pro přístup k datům v různých datových zdrojích z aplikací pro .NET framework. Typy spojené s komponentou ADO.NET (obr. 9.1) se nacházejí ve jmenných prostorech rozšiřující jmenný prostor System.Data (System.Data.Common, System.Data.SqlClient, System.Data.Odbc apod.).“ [10]

Použití různých datových zdrojů

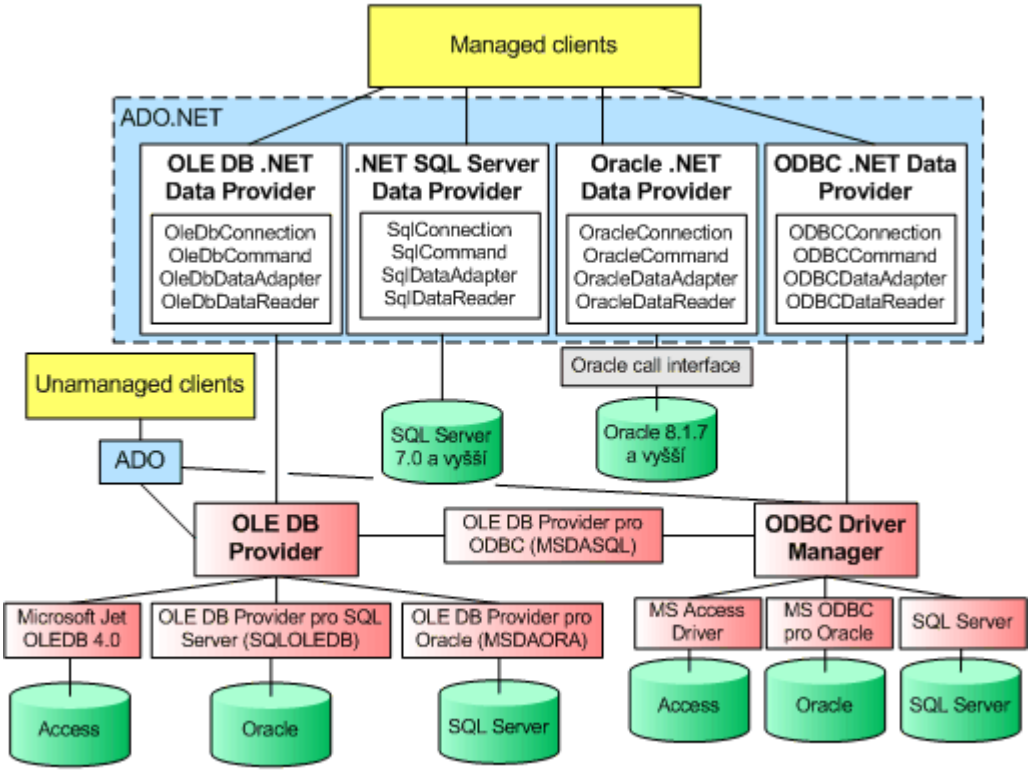
Možnost přístupu k datům v různých datových zdrojích je v technologii ADO.NET [10] zajištěna tak, že jsou předepsána určitá rozhraní (*IDbConnection, IDbCommand, IDataReader, IDataAdapter* apod.). Tato rozhraní předepisují funkčnost, kterou musí splňovat jednotliví zprostředkovatelé přístupu ke konkrétnímu datovému zdroji, pro nějž je zprostředkovatel implementován. Tito zprostředkovatelé jsou oficiálně nazýváni ADO.NET data providers. Použitím přístupu pomocí rozhraní je dosaženo toho, že je možné psát takový kód, který bude fungovat pro jakýkoli datový zdroj. To znamená, že pokud se změní datový zdroj naší aplikace (např. databáze MS SQL na databázi Oracle) a naše aplikace je napsána s použitím přístupu přes zmíněná obecná rozhraní ADO.NET, chod aplikace zůstane stejný. Základní knihovna tříd .NET frameworku obsahuje několik základních ADO.NET data providerů, které obsahují třídy implementující výše zmíněná rozhraní, a to pro následující konkrétní datové zdroje:

- Databáze Microsoft SQL Server (7.0 nebo vyšší) – jmenný prostor *System.Data.SqlClient*
- ODBC datové zdroje – jmenný prostor *System.Data.Odbc*
- OLE DB datové zdroje – jmenný prostor *System.Data.OleDb*
- Databáze Oracle – jmenný prostor *System.Data.Oracle*

Já používám ve své implementaci ODBC datový zdroj, tedy jmenný prostor *Systém.Data.Odbc*. Vzhledem k tomu, že aplikaci bylo nutno zkoušet na nějaké konkrétní databázi, nainstaloval jsem si MySQL ODBC Provider (mysql-connector-odbc-3.51.12).

Online a offline aplikace

„*Ke komunikaci aplikace s nějakým datovým zdrojem můžeme použít dva scénáře pro přístup k datům, a to scénář, kdy je aplikace takzvaně připojena k datovému zdroji (online), nebo naopak je od datového zdroje většinu času odpojena (offline). Komponenta ADO.NET je navržena tak, aby bylo možné ji použít v obou těchto scénářích a já také oba scénáře ve své implementaci potřebuji.*“ [10]



Obr. 9.1 Komponenta ADO.NET [15]

Online aplikace

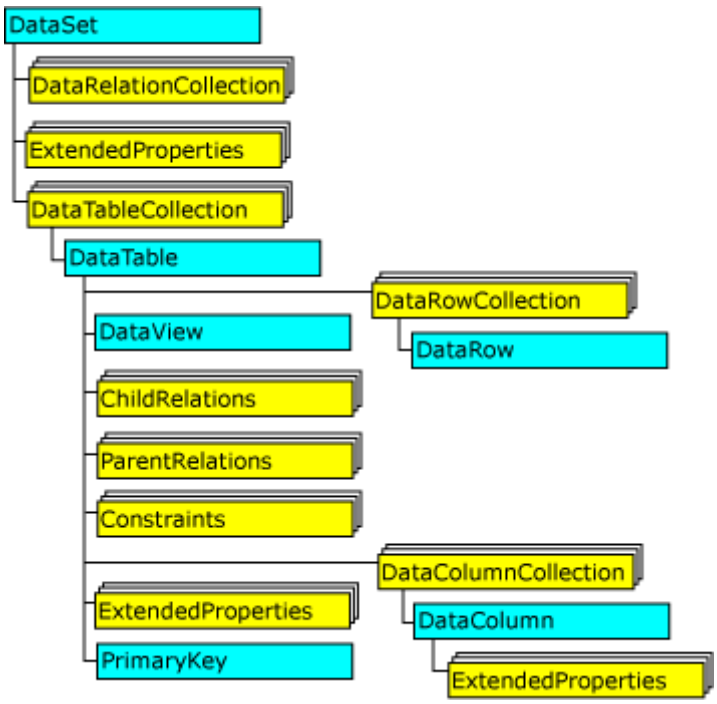
Tento typ aplikace má vždy při manipulaci se souvisejícím datovým zdrojem aktivní připojení [10]. To je vhodné v případech, kdy potřebujeme přistupovat k datům, která jsou často měněna. Na druhou stranu to může představovat nevýhodu v podobě zvýšené komunikace mezi aplikační vrstvou a vrstvou datovou (tedy mezi aplikací a například databází). V případě ADO.NET jsou tyto scénáře řešeny pomocí kombinace jednotlivých implementací rozhraní *IDbConnection, IDbCommand* (v případě potřeby v souvislosti s implementací rozhraní *IDbParameter*) a *IDataReader* pro konkrétní datový zdroj.

Offline aplikace

Druhým způsobem je použití odpojeného datového zdroje. V tomto případě jsou data z tohoto datového zdroje získána do aplikace a poté je spojení s datovým zdrojem ukončeno. Od chvíle odpojení od datového zdroje aplikace manipuluje s obrazem získaných dat, který je uložen v paměti.
Po dokončení potřebných modifikací dat na úrovní aplikační logiky je opět vytvořeno aktivní spojení k datovému zdroji, zjištěny rozdíly mezi daty z aplikace a daty v datovém zdroji a poté jsou tyto rozdíly promítnuty do příslušného datového zdroje. Technologie ADO.NET má pro toto řešení výbornou podporu, a to v podobě třídy *DataSet* (obr. 9.2), která představuje onen obraz získaných

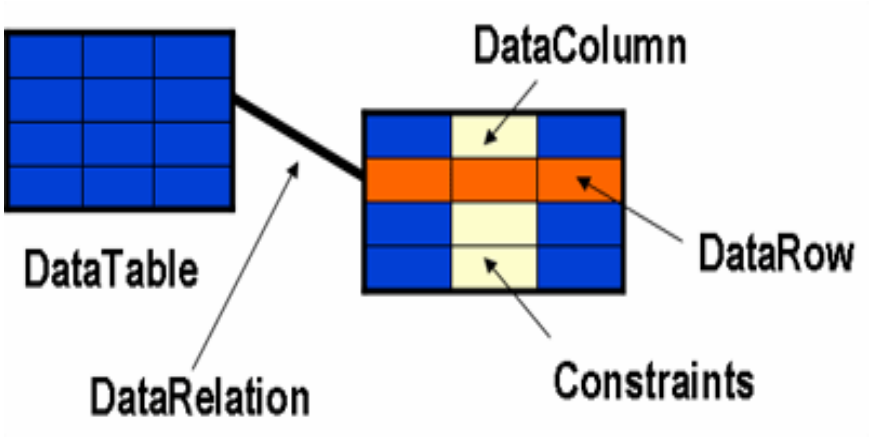
dat v paměti. Instance této třídy je naplněna daty pomocí konkrétní implementace rozhraní *IDataAdapter*, který využívá implementace rozhraní *IDbConnection* a *IDbCommand*. K tomu, aby mohl *DataSet* vystupovat jako náhražka databáze, disponuje schopnostmi uchovávat obrazy jedné nebo více tabulek. Je možné pro jednotlivé tabulky definovat různá omezení (constraints), nebo také různé relace mezi jednotlivými tabulkami v *DataSetu*, což ve výsledku přináší opravdu velmi dobrou podporu pro vytvoření obrazu databáze pro práci offline. Narozdíl od implementace online aplikací v ADO.NET, kde se pro čtení dat z datového zdroje používaly implementace rozhraní *IDataReader*, při jejichž používání je možné číst data pouze jedním směrem, je v případě využití třídy *DataSet* a s ním spojených tříd možno číst data bez jakýchkoli takovýchto omezení.

„Ovšem při implementaci přístupu k datům použitím třídy DataSet nejsme omezeni pouze na čtení dat, ale je možné s daty provádět i změny, které jsou následně při připojení k datovému zdroji do něj promítnuty. Také na rozdíl od dříve probíraného připojeného scénáře je možné, aby DataSet obsahoval reprezentaci dat z různých datových zdrojů, tedy že nějaké tabulky budou z jednoho datového zdroje a další tabulky zase z jiného datového zdroje. K reprezentaci databázové tabulky v instanci třídy DataSet slouží třída DataTable (obr. 9.3), jejíž instance se nacházejí v kolekci DataTableCollection, představovanou vlastností Tables daného DataSetu. K vyjádření relací mezi objekty typu DataTable je k dispozici třída DataRelation a její instance jsou obsaženy v kolekci typu DataRelationCollection, kterého je vlastnost Relations třídy DataSet.“ [10]



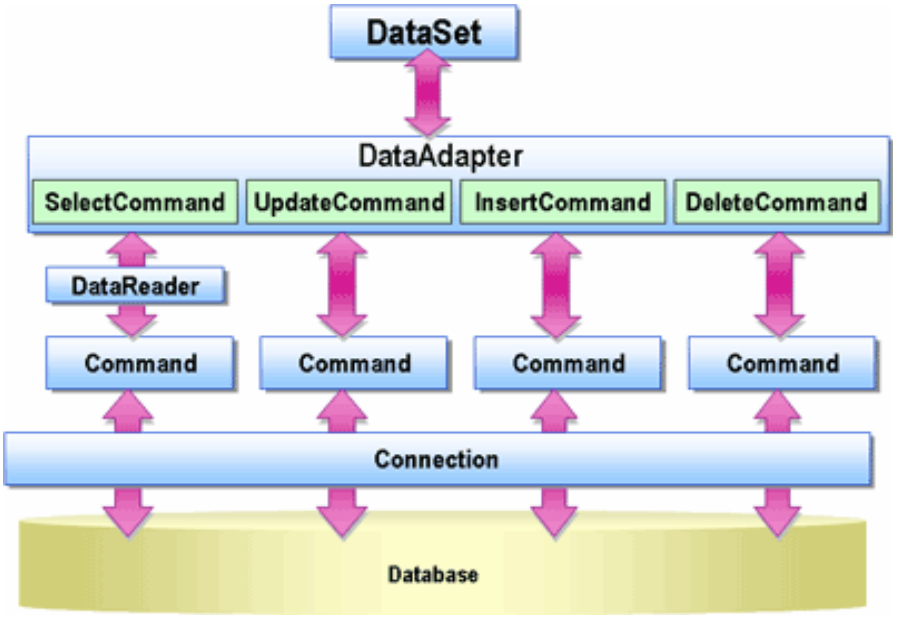
Obr. 9.2 Schéma objektu DataSet [10]

Tabulka ze zdroje je tedy v paměti reprezentována objekty typu *DataTable*. Stejně jako v databázi, tak i reprezentace tabulky v podobě objektu *DataTable* obsahuje sloupce různých datových typů. Jednotlivé sloupce jsou reprezentovány objekty typu *DataColumn*. Tyto reprezentace sloupců jsou obsaženy v kolekci *DataColumnCollection*. Tuto kolekci má každý objekt *DataTable* přístupnou skrze svou vlastnost *Columns*. Tak jako sloupce jsou i jednotlivé řádky v tabulce, představující záznamy, reprezentovány objekty. Tyto objekty jsou typu *DataRow* a nacházejí se v kolekci *DataRowCollection*, přístupnou pomocí vlastnosti *Rows* objektu *DataTable*.



Obr. 9.3 Schéma objektu DataTable [10]

Úlohu prostředníka mezi objektem *DataSet* a vlastním datovým zdrojem mají instance tříd implementujících rozhraní *IdataAdapter* (obr. 9.4). Toto rozhraní je stejně jako například v případě rozhraní *IDbConnection* implementováno na úrovni jednotlivých ADO.NET data providerů a umožňuje naplnit *DataSet* daty z datového zdroje a také promítnout změny, které se provedly s daty v objektu *DataSet*, zpět do datového zdroje. Konkrétní implementace typu *IDataAdapter* má asociované spojení k datovému zdroji a toto spojení využívá k získávání či úpravě dat. To, jaká data jsou získána nebo jaký záznam je přidán/upraven/odebrán, je dáno definicí jednotlivých příkazů, které jsou představovány typem *IDbCommand*.



Obr. 9.4 Funkce datového adaptéru [10]

Načtení dat

Některé údaje později vkládané do databáze jsou načítány přímo v průběhu běhu aplikace, některé je ovšem nutno načíst ze souboru. Jsou to jednotlivé názvy vzorků s jejich fragmenty. Vzhledem k tomu, že jsou data tabulkového formátu, vyskytují se výhradně ve formátu Microsoft Office Excel (*.xls). Je důležité najít cestu, jak data tohoto formátu načíst do naší aplikace. Zvolil jsem formát XML (Extensible Markup Language) pro jeho univerzálnost a jasnou strukturu, ze které se dají data konečným automatem zpracovat. Microsoft Office Excel již obsahuje utilitu na uložení zobrazených dat ve formátu XML.

Zobrazení dat

Velmi důležitou částí implementace je vyřešit, jak budou data z databáze (potažmo z *DataSetu*) zobrazena. K tomuto účel jsem užil *DataGridView*. „*Ovládací prvek DataGrid, který je k dispozici od první verze platformy .NET, byl sice funkční, ale neumožňoval mimo jíné zobrazit obrázky či rozevírací ovládací prvky nebo uzamknout sloupce. Na platformě 2.0 a výše je k dispozici DataGridView, který řeší mnoho nedostatků původního ovládacího prvku*.“ [11] Je ovšem nutné vypořádat se se skutečností, že počet sloupců této komponenty je omezen. Vyhledal jsem na foru společnosti Microsoft, že neoptimalizovali tuto komponentu pro více než 100 sloupců. Dalším hledáním a zkoušením v aplikaci jsem zjistil, že maximální podporovaný počet sloupců komponenty DataGridView je 750, přičemž zobrazení nemusí řádně fungovat pro více než 300 sloupců. Proto s tím v implementaci aplikace musíme počítat a dovolit zobrazení pouze 300 sloupců.

Práce s maticí

Matice nemůže být tvořena staticky, jelikož je nutné do ní prvky vkládat i z ní odebírat v průběhu výpočtů. Datový typ dvojrozměrné pole je tedy nevyhovující. Využil jsem nabízených typů v C# a naimplementoval jsem matici jako seznam seznamů, přesněji *List ArrayListů*. *ArrayList* vychází z datového typu *Array* (pole), ale má dynamický charakter, takže poskytuje metody pro vkládání a mazání prvků na jakékoli pozici. Navíc má *ArrayList* úzkou vazbu s komponentou *DataGridView*, takže z matice mohou být připravena jednodušeji k zobrazení. Je tu ovšem problém se zobrazením správných hodnot (typů) dat v objektu DataGridView. „*Místo řetězců definovaných v poli se v mřížce zobrazí délka těchto řetězců. Je to způsobeno tím, že při použití pole jako zdroje dat pro ovládací prvek typu DataGridView mřížka vyhledá první veřejnou vlastnost objektu – v našem případě pole – a zobrazí místo řetězcové hodnoty tuto veřejnou vlastnost. Jeden ze způsobů, jak vyřešit problém zobrazení řetězců v DataGridView, spočívá ve vytvoření obalové třídy*.“ [11] Upravil jsem obalovou třídu z CodeProject [12], která umožňuje správné zobrazení 2-dimenzionálního pole.

Výsledky

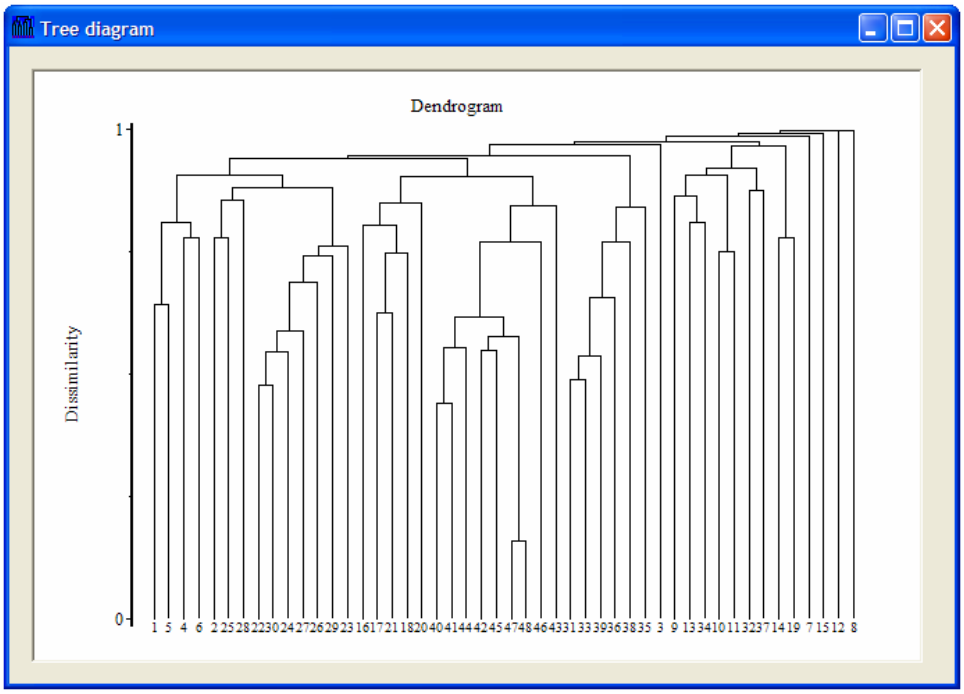
Po provedení všech potřebných kroků a získání matice podobnosti je dosaženo cíle. Matice podobnosti se ukládá do souboru *.dis (distance matrix) ve formátu připraveném pro zpracování programem Syntax 2000. Zvážena byla možnost zobrazit výsledné dendrogramy či scattergramy přímo v aplikaci, ale kvůli nedostupnosti knihoven a prostředků zdarma byla tato možnost zamítnuta. Některými z produktů, které nabízejí knihovny pro zobrazení dendrogramů a scattergramů, jsou balík IMSL od Visual Numerics[13] a ArcGIS od ESRI[14].

10 Testy

V této části ukážeme, jaké výsledky aplikace vyhodnotila na různě zadaný rozsah přesnosti (tolerance blízkosti) a různé koeficienty. Testy samotného chování aplikace zde nebudou zapsány, chování aplikace je popsáno v uživatelské příručce.

Budeme vycházet ze souboru pěti druhů, které jsou v databázi uloženy (jsou označeny jako: A, M, O, S a V). Každý z druhů byl rozkapán třemi barvami – B, G a Y. Do databáze jsou v pořádku načteny fragmenty k jednotlivým druhům. Soubory XML se vzorky obsahující fragmenty k jednotlivým druhům (rozkapanými jednotlivými barvami) jsou dodávány společně s aplikací jako testovací data (zde u jednotlivých testů neuváděny vzhledem k rozsáhlosti dat) na CD.

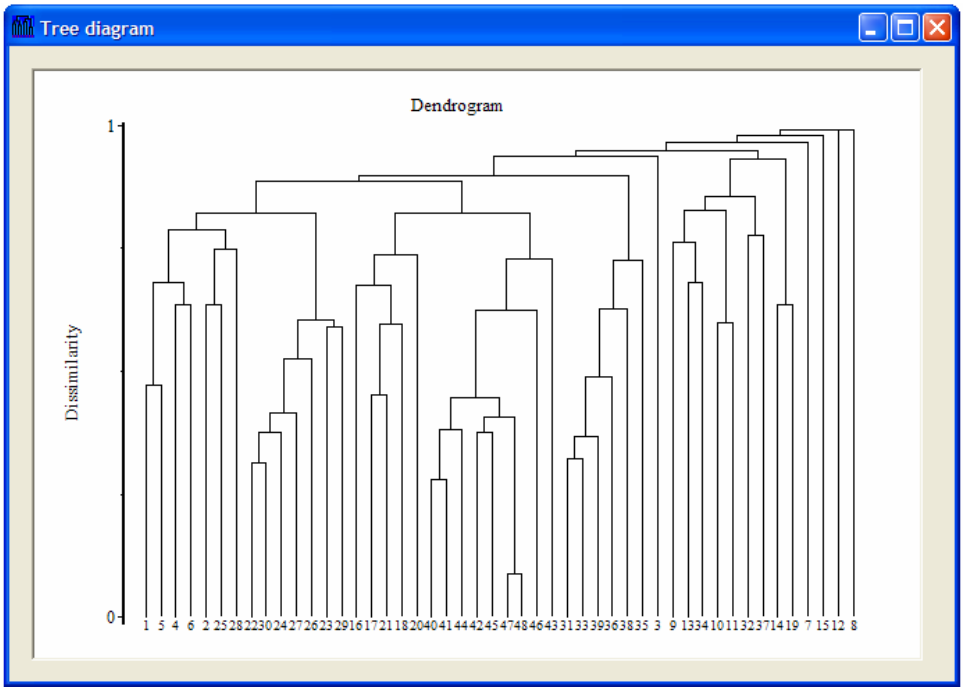
Test 1:
Srovnání: A: B,G,Y M: B,G,Y
Tolerance blízkosti: 0.65; Koeficient: Jaccard; Pro vykreslení užita metoda: UPGMA



Test 2:

Srovnání: A: B,G,Y M: B,G,Y

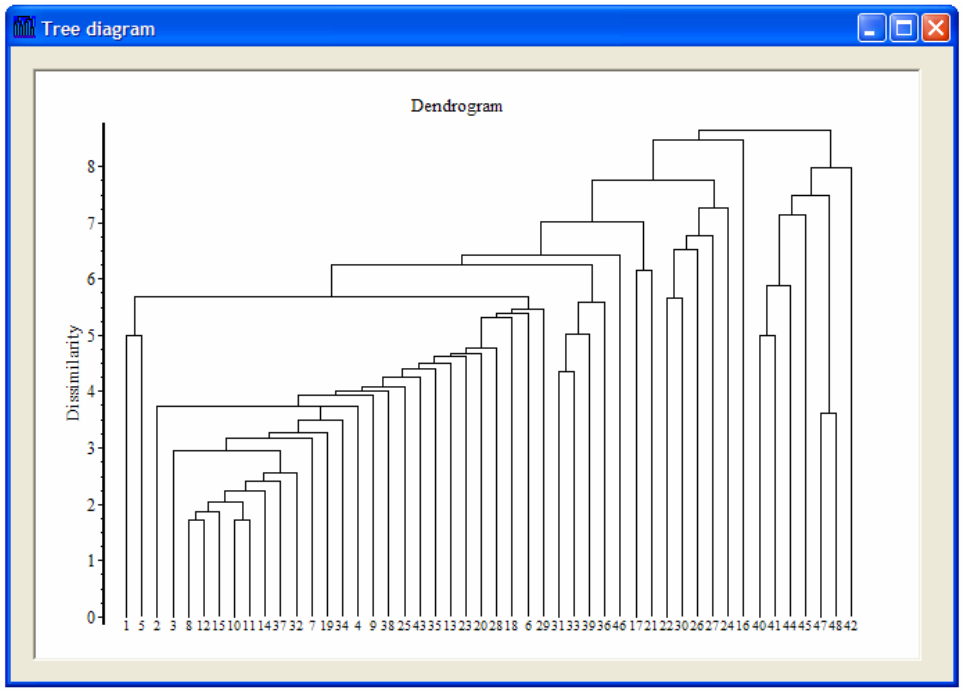
Tolerance blízkosti: 0.65; Koeficient: Sörensen; Pro vykreslení užita metoda: UPGMA



Test 3:

Srovnání: A: B,G,Y M: B,G,Y

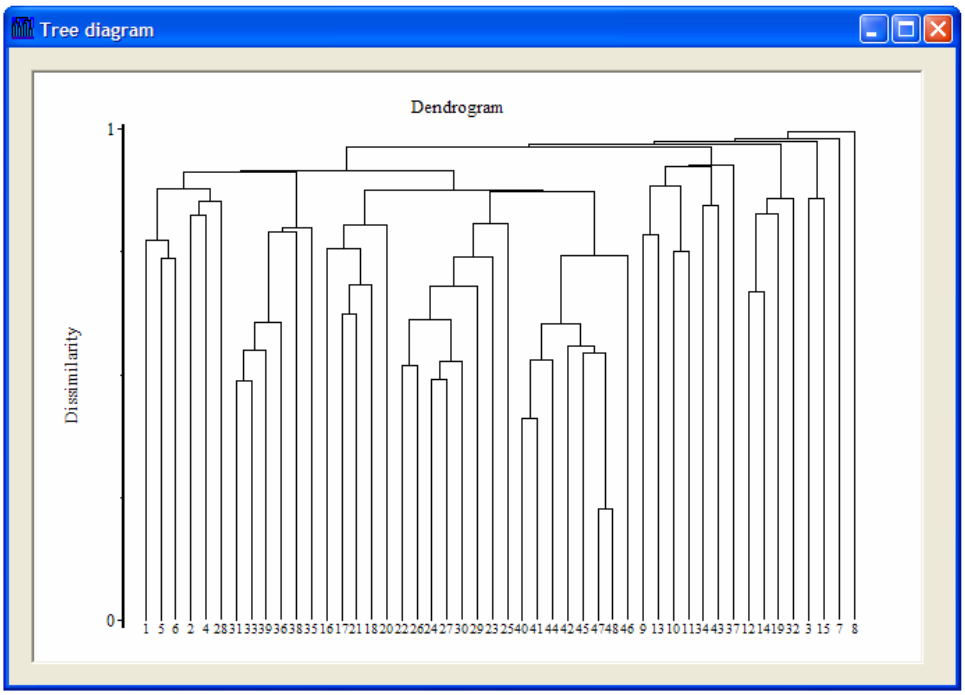
Tolerance blízkosti: 0.65; Koeficient: Euklidovský; Pro vykreslení užita metoda: UPGMA



Test 4:

Srovnání: A: B,G,Y M: B,G,Y

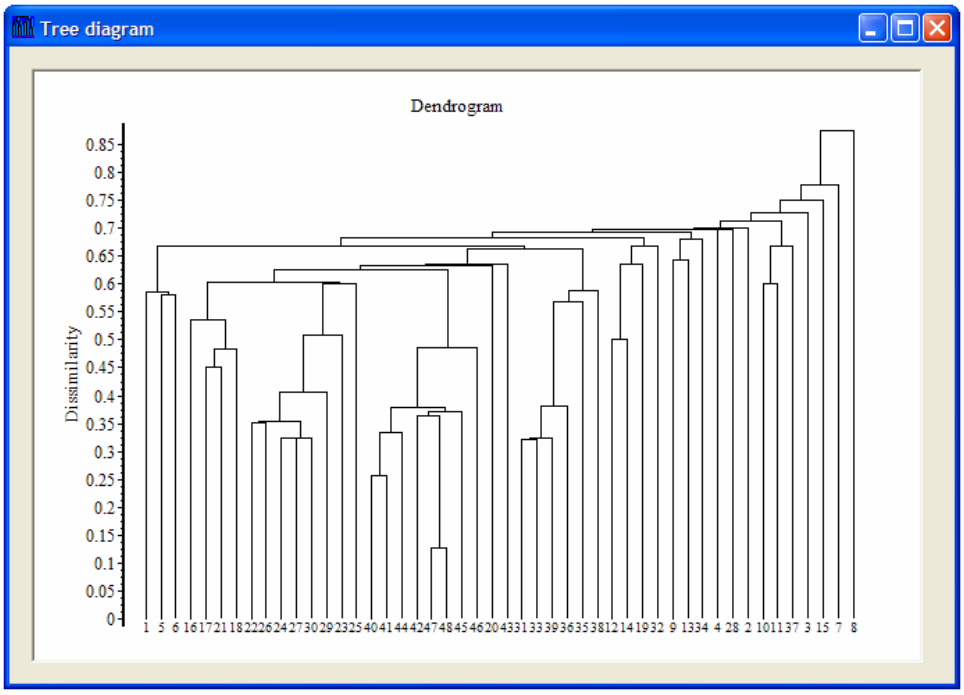
Tolerance blízkosti: 1.0; Koeficient: Jaccard; Pro vykreslení užita metoda: UPGMA



Test 5:

Srovnání: A: B,G,Y M: B,G,Y

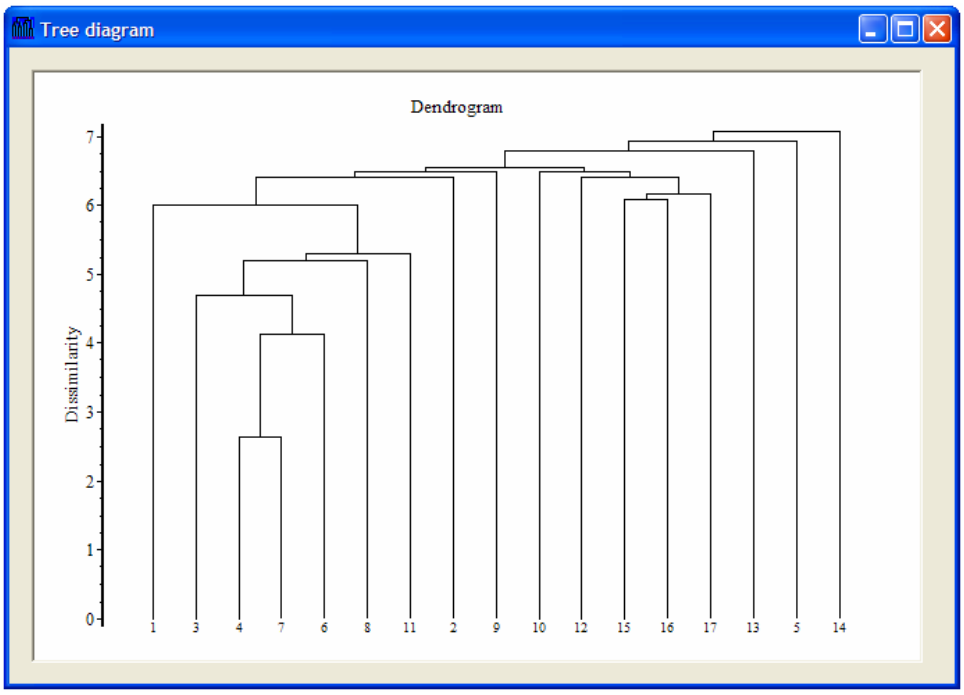
Tolerance blízkosti: 1.0; Koeficient: Sörensen; Pro vykreslení užita metoda: Single Linkage



Test 6:

Srovnání: S: Y V: Y

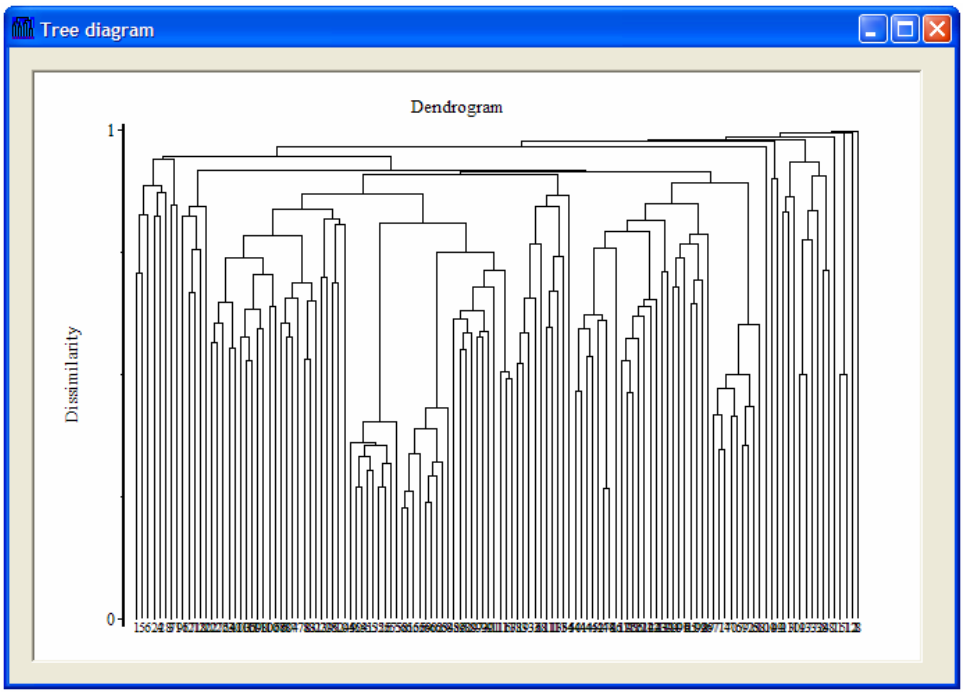
Tolerance blízkosti: 0.65; Koeficient: Euklidovský; Pro vykreslení užita metoda: Single Linkage



Test 6:

Srovnání: A: B,G,Y M: B,G,Y O: B,G,Y S: B,G,Y V: B,G,Y

Tolerance blízkosti: 0.65; Koeficient: Jaccard; Pro vykreslení užita metoda: UPGMA



11 Závěr

Ve výše uvedených kapitolách jsme se postupně seznámili s genovým inženýrstvím, makromolekulou DNA, metodami pro purifikaci a separaci nukleových kyselin, enzymy používanými k úpravám těchto kyselin a amplifikací. Zabývali jsme se také teoretickou částí shlukové a gradientové analýzy včetně koeficientů statistických metod, existujícími nástroji pro tuto oblast, navrhli jsme vlastní aplikaci a naimplementovali jsme ji v objektovém programovacím nástroji Microsoft Visual Studio 2005. V předposlední kapitole jsme si ukázaly vybrané testy, které byly zkonzultovány se zadavatelem projektu Mgr. Františkem Zedkem z Přírodovědecké fakulty Masarykovy university Brno. Aplikace byla zadavatelem vyhodnocena jako úspěšná. Byl projednán další možný vývoj, a to zejména možnost vytvořit aplikaci jako webovou. Taktéž by bylo vhodné, aby aplikace uměla zobrazit výsledné dendrogramy a scattergramy.

Literatura

[1] Jaša, P. *Techniky pro získávání dat v genomice : Diplomová práce.* Brno : VUT Brno, Fakulta informačních technologií, 2007.

[2] Šmarda, J.; Doškař, J.; Pantůček, R.; Růžičková, V.; Koptíková, J. *Metody molekulární biologie : učební skriptum.* Brno : Masarykova univerzita Brno, 2005.

[3] Wikipedia contributors. *Wikipedia, The Free Encyclopedia : DNA* [online], [cit. 2007-11-22]. Dostupný z WWW: <http://en.wikipedia.org/wiki/DNA>

[4] Exeter Software. *SYN-TAX 2000 : Data Analysis in Ecology and Systematics* [online], [cit. 2007-12-19].
Dostupný z WWW: <http://www.exetersoftware.com/cat/syntax/syntax.html>

[5] Marhold, K. *Botanická informatika : přednáška* [online]. 2006, [cit. 2008-05-12]. Bratislava : Botanický ústav SAV.
Dostupný z WWW: <http://www.ibot.sav.sk/karolx/Prednasky/ZaFeKla3.ppt>

[6] Řezanková, H. *Klasifikace pomocí shlukové analýzy : přednáška* [online]. 2003, [cit. 2008-05-13]. Praha : Vysoká škola ekonomická v Praze.
Dostupný z WWW: <http://nb.vse.cz/~rezanka/Shlukova_analyza2003.pdf>

[7] Ježek, K. *Datamining : přednáška* [online]. 2005, [cit. 2008-05-12]. Plzeň : Západočeská universita v Plzni, Fakulta aplikovaných věd, Katedra informatiky a výpočetní techniky.
Dostupný z WWW:
<http://www.kiv.zcu.cz/~jezek_ka/vyuka/DB2%202005/Datamining/Shlukov%E1n%ED.doc>

[8] Statsoft. *Statistica : Analýza dat* [online], [cit. 2008-05-13].
Dostupný z WWW: <http://www.statsoft.cz/page/index2.php?pg=navigace&nav=12>

[9] Hatina, P. PC Svět. *Programovací jazyk C# (1.díl)* [online], [cit. 2008-05-12].
Dostupný z WWW: <http://www.pcsvet.cz/art/article.php?id=4848>

[10] Puš, P. Computer Press, a. s. *Poznáváme C# a Microsoft .NET (54. a 60. díl)* [online], [cit. 2008-05-14].
Dostupný z WWW: <http://www.zive.cz/Titulni-strana/Poznavame-C-a-Microsoft-NET--54-dil--ADO-NET/sc-21-a-128344/default.aspx>

[11] Nagel, Ch.; Evjen, B.; Glynn, J.; Skinner, M.; Watson, K.; Jones A. *C# 2005, Programujeme profesionálně.* Vyd.1. Brno : Computer Press, 2006. 1398 s. ISBN 80-251-1181-4.

[12] Stefanov, M. The Code Project. *Binding a two dimensional array to a DataGrid* [online], [cit. 2008-05-12].
Dostupný z WWW: <http://www.codeproject.com/KB/database/BindArrayGrid.aspx>

[13] Visual Numerics. *IMSL C# Programmer's Guide* [online], [cit. 2008-05-10]. Dostupný z WWW: <http://www.vni.com/products/imsl/cSharp/v50/manual/chartpg/dendrogram.html>

[14] ESRI. *ArcGIS 9.2 - How Dendrogram works* [online], [cit. 2008-05-12]. Dostupný z WWW: <http://webhelp.esri.com/arcgisdesktop/9.2/index.cfm?TopicName=How%20Dendrogram%20works>

[15] Beňo, M. Devmasters s.r.o. *Úvod do ADO.NET (1.díl)* [online], [cit. 2008-05-12]. Dostupný z WWW: <http://www.vyvojar.cz/Articles/371-uvod-do-ado-net-1.aspx>

[16] Pirkl, J. *Řešené příklady v C# aneb C# skutečně prakticky*. České Budějovice : Kopp, 2005. 300 s. ISBN 80-7232-265-6.

[17] Price, J. *C# programování databází*. Praha : Grada, 2005. 623 s. ISBN 80-247-0982-1.

Seznam příloh

Příloha I – Uživatelská příručka

Úvod

Tento program byl vytvořen jako součást diplomové práce. Jedná se o aplikaci srovnávající různé druhy rostlin. Děje se tak na základě vzorků jednotlivých druhů uložených v databázi. Uživatel má možnost nové vzorky do databáze uložit i z ní vybrat ty, které chce srovnat.

Instalace potřebných součástí

Aplikace je vytvořená na platformě .NET, tudíž je ke správné funkci programu potřeba nainstalovat *Microsoft .NET Framework verze 2.0 či vyšší*. Je volně ke stažení na internetu (například přímo na stránkách firmy Microsoft). Framework není třeba instalovat, pokud jste si již dříve nainstalovali *.NET Framework SDK* či *Microsoft Visual Studio .NET*.

K databázi bude aplikace přistupovat přes rozhraní ODBC. Potřeba je tedy nainstalovat *ODBC driver* (ovladač). Instalace *ODBC driveru* pro přístup k databázím *MySQL Serveru* je dodávána společně s touto aplikací na přiloženém CD.

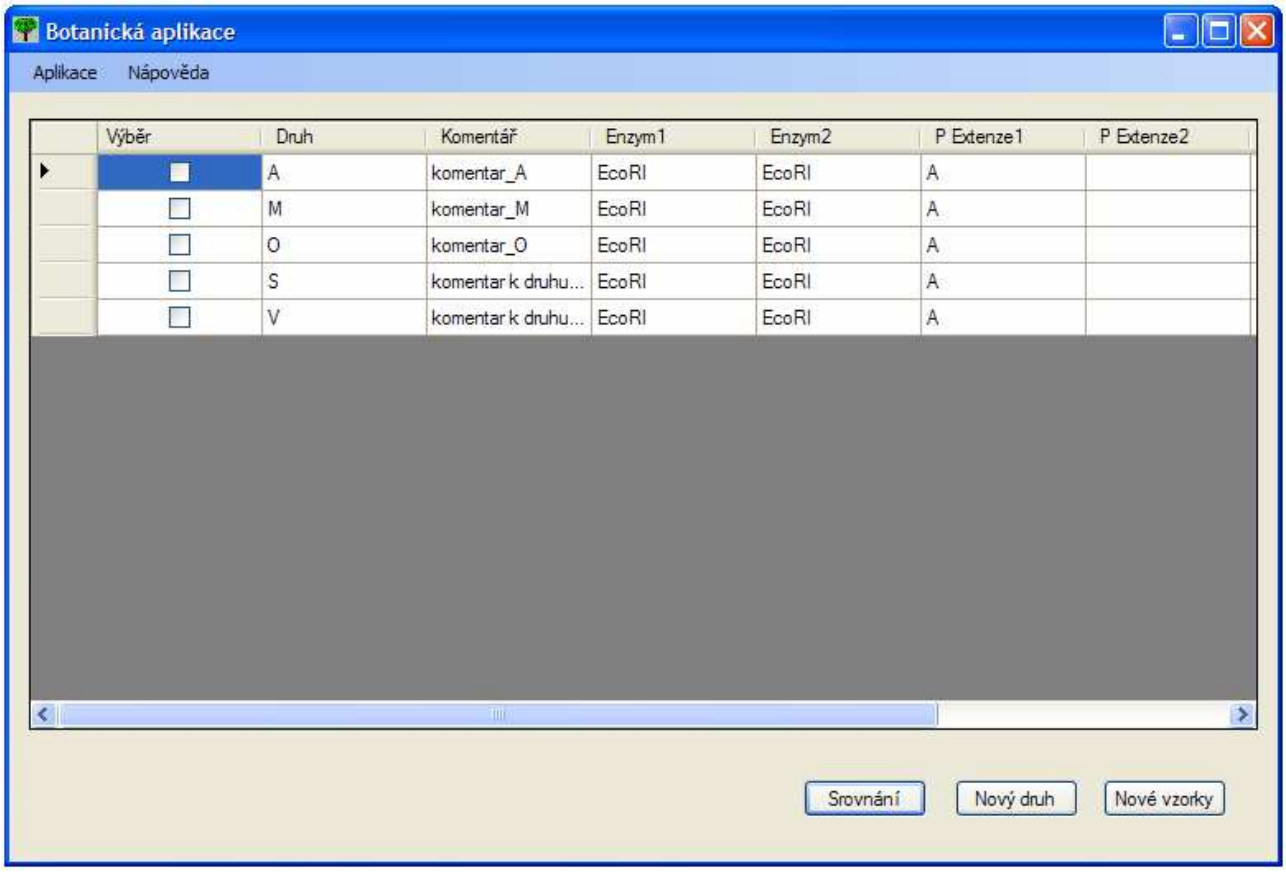
Práce s programem

Po úspěšné instalaci potřebných součástí systému spusťte pro start programu soubor „Botanic.exe“. Aplikace spustí přihlašovací formulář, kde je nutné vyplnit login, heslo a adresu databázového serveru.

Po zadání přihlašovacích údajů se aplikace ihned zkusí spojit s databází a v případě, že se nepodařilo spojení navázat, objeví se nápis varující, že se nepodařilo připojit k databázi a je nutno zadat přihlašovací údaje znovu.

Pokud se podařilo spojení s databází navázat, zobrazí se hlavní okno aplikace obsahující tabulku s dostupnými druhy v databázi. Jsou platná všechna tlačítka a aplikace je připravena na další akci (obr. 3.1).

Menu aplikace obsahuje pouze dvě položky. V položce *Nápověda* je dostupná volba *O aplikaci* a v položce *Aplikace* volbou *Konec* se program ukončí.



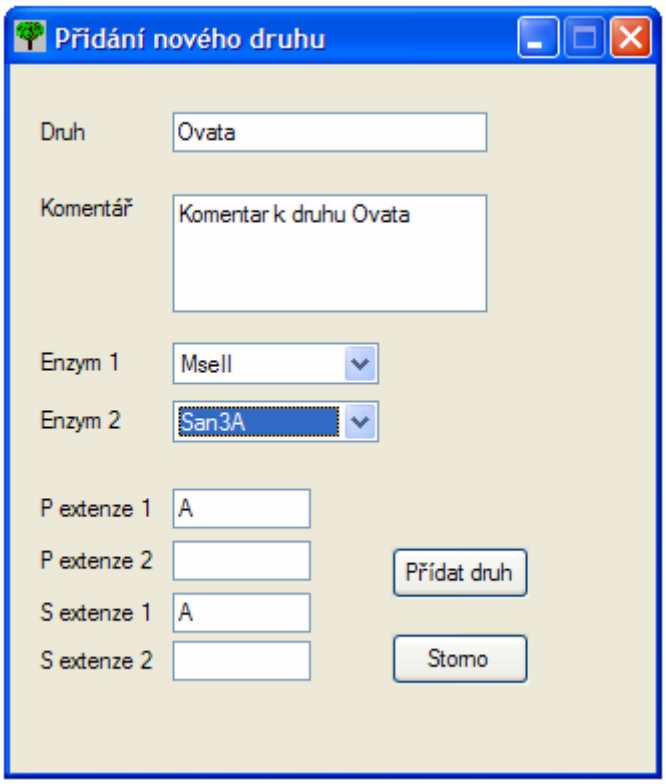
Obr. 3.1 Hlavní okno aplikace po přihlášení

Přidání nového druhu

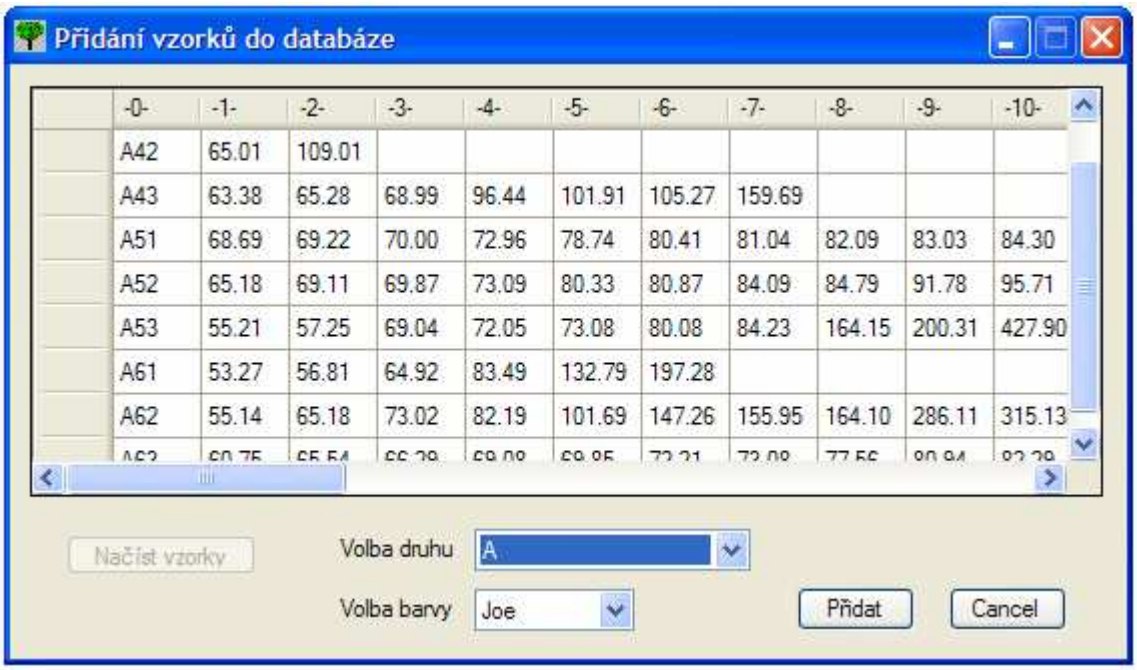
Jestliže uživatel spustí tlačítko *Nový druh*, otevře se formulář (obr. 3.2) pro zadání jednotlivých údajů o novém druhu. Jde o název druhu, komentář, volbu dvou enzymů, zadání extenzí v preselekci i v selekci. Potvrzení přidání nového druhu se děje tlačítkem *Přidat druh*. Tlačítkem *Storno* zrušíme zadávání druhu.

Přidání nových vzorků

Po spuštění tlačítka *Nové vzorky* se otevře nový prázdný formulář s platným tlačítkem *Načíst vzorky*. Po spuštění tlačítka *Načíst vzorky* je umožněno uživateli vybrat soubor ve formátu XML, ve kterém mají být vzorky uloženy. Po správném načtení vzorků ze souboru formátu XML se zneplatní tlačítko *Načíst vzorky* a na formuláři se objeví možnosti *Volba druhu* a *Volba barvy* společně s tlačítkem *Přidat* (obr. 3.3). Zvolíme tedy druh a barvu, zkontrolujeme načtená data (název vzorku v prvním sloupci následovaný délkami fragmentů v ostatních sloupcích) a potvrdíme zmíněným tlačítkem *Přidat*. Po této akci se nahrají nová data do databáze a jsou připravena k použití.



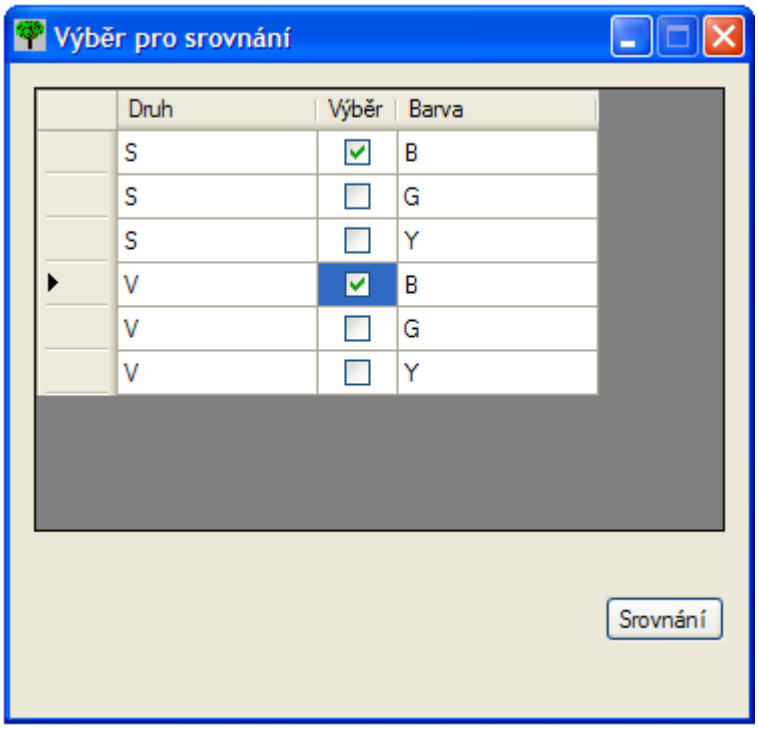
Obr. 3.2 Přidání nového druhu



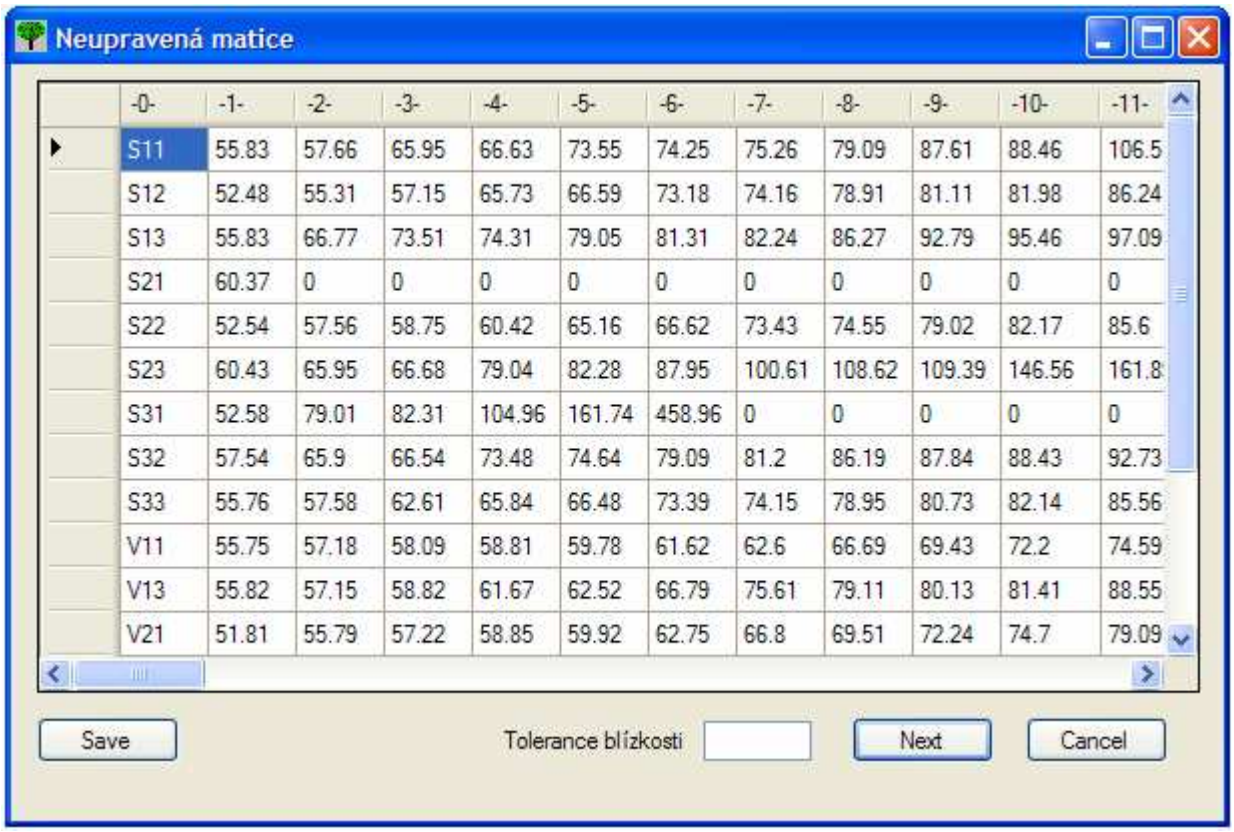
Obr. 3.3 Přidání vzorků do databáze

Srovnání vzorků

Pro srovnání vzorků druhů musí uživatel označit v hlavním okně druhy, které chce srovnávat. Pak spustí tlačítkem *Srovnání* nové okno, kde se mu zobrazí označené druhy navíc s jim příslušejícími barvami, kterými byly rozkapány a jejichž vzorky jsou zadány v databázi (obr. 3.4). Tam tedy uživatel označí, které vzorky bude chtít srovnávat, a potvrdí tlačítkem *Srovnání*. Objeví se nové okno s neupravenou maticí tvořenou jednotlivými vzorky (skládající se z fragmentů, obr. 3.5). Uživatel zadá toleranci blízkosti fragmentů a tlačítkem *Další* (což značí další krok) přejde okno do stavu zobrazení upravené matice (obr. 3.6). Implicitně je nastavena tolerance blízkosti 0.65. Dalším použitím tlačítka *Další* se provede úprava upravené matice na matici přítomnosti a nepřítomnosti fragmentů (obr. 3.7). Nyní lze zvolit koeficient podobnosti (nepodobnosti). Implicitně je zvolený Jaccardův koeficient. Opětovným použitím tlačítka *Další* se vypočítá a zobrazí matice podobnosti, což je výsledek (obr. 3.8). V této chvíli je doporučeno použitím tlačítka *Uložit* výslednou matici uložit na disk, odkud bude moci být koeficient vyvolán například programem Syntax 2000 a zobrazen výsledek formou grafu rozptýlení. V dřívějších krocích úpravy matice lze matici samozřejmě také ukládat. Vždy je taktéž přítomné tlačítko *Storno* pro zrušení dané akce.

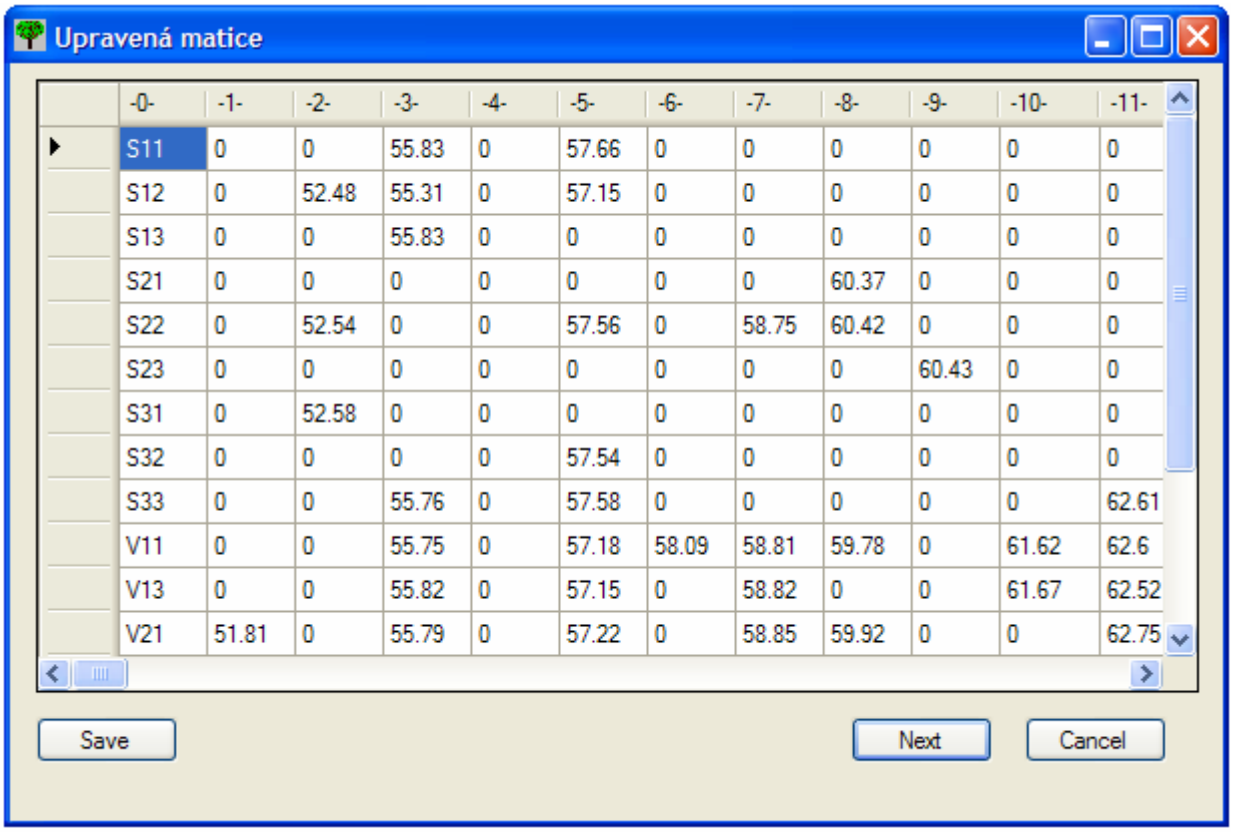


Obr. 3.4 Výběr druhů a barev pro srovnání



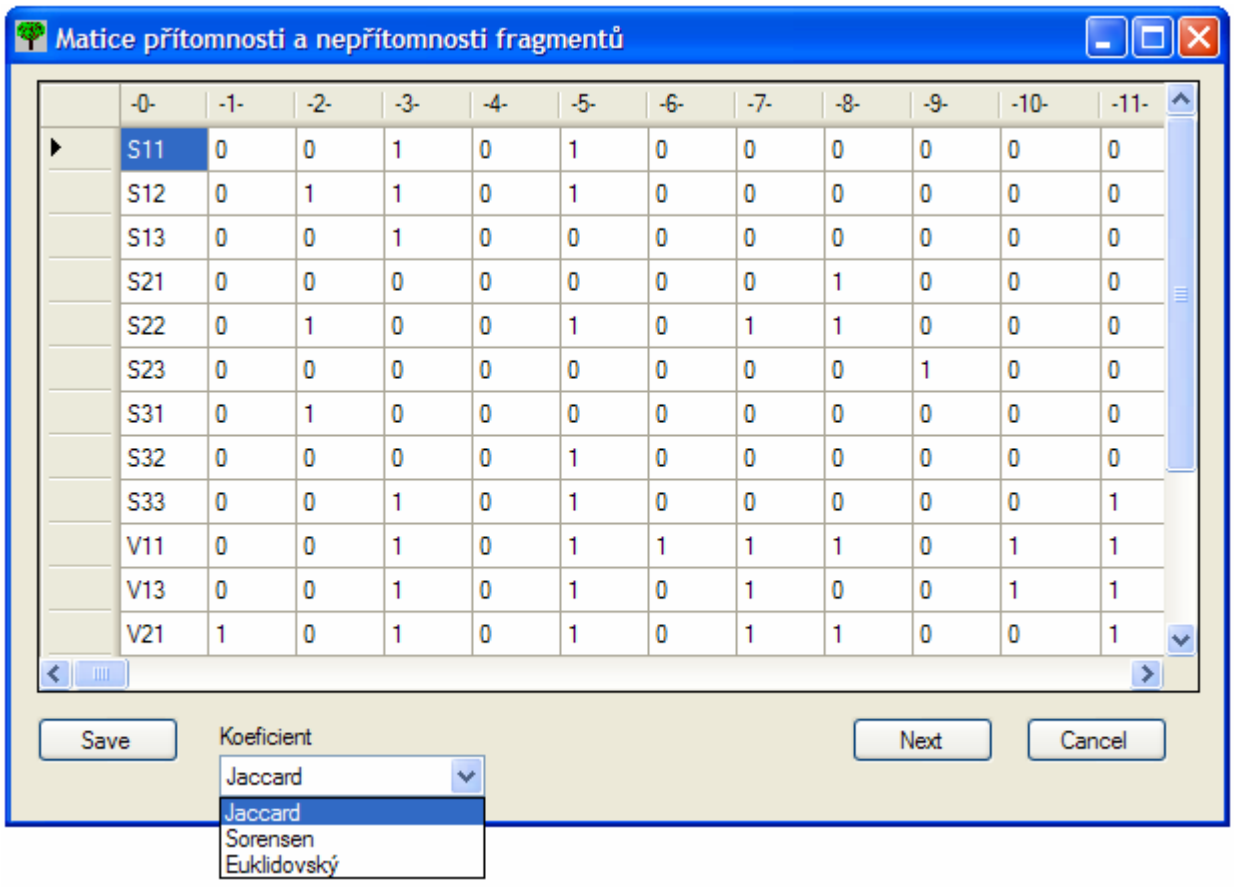
	-0-	-1-	-2-	-3-	-4-	-5-	-6-	-7-	-8-	-9-	-10-	-11-
▶	S11	55.83	57.66	65.95	66.63	73.55	74.25	75.26	79.09	87.61	88.46	106.5
	S12	52.48	55.31	57.15	65.73	66.59	73.18	74.16	78.91	81.11	81.98	86.24
	S13	55.83	66.77	73.51	74.31	79.05	81.31	82.24	86.27	92.79	95.46	97.09
	S21	60.37	0	0	0	0	0	0	0	0	0	0
	S22	52.54	57.56	58.75	60.42	65.16	66.62	73.43	74.55	79.02	82.17	85.6
	S23	60.43	65.95	66.68	79.04	82.28	87.95	100.61	108.62	109.39	146.56	161.8
	S31	52.58	79.01	82.31	104.96	161.74	458.96	0	0	0	0	0
	S32	57.54	65.9	66.54	73.48	74.64	79.09	81.2	86.19	87.84	88.43	92.73
	S33	55.76	57.58	62.61	65.84	66.48	73.39	74.15	78.95	80.73	82.14	85.56
	V11	55.75	57.18	58.09	58.81	59.78	61.62	62.6	66.69	69.43	72.2	74.59
	V13	55.82	57.15	58.82	61.67	62.52	66.79	75.61	79.11	80.13	81.41	88.55
	V21	51.81	55.79	57.22	58.85	59.92	62.75	66.8	69.51	72.24	74.7	79.09

Obr. 3.5 Neupravená matice

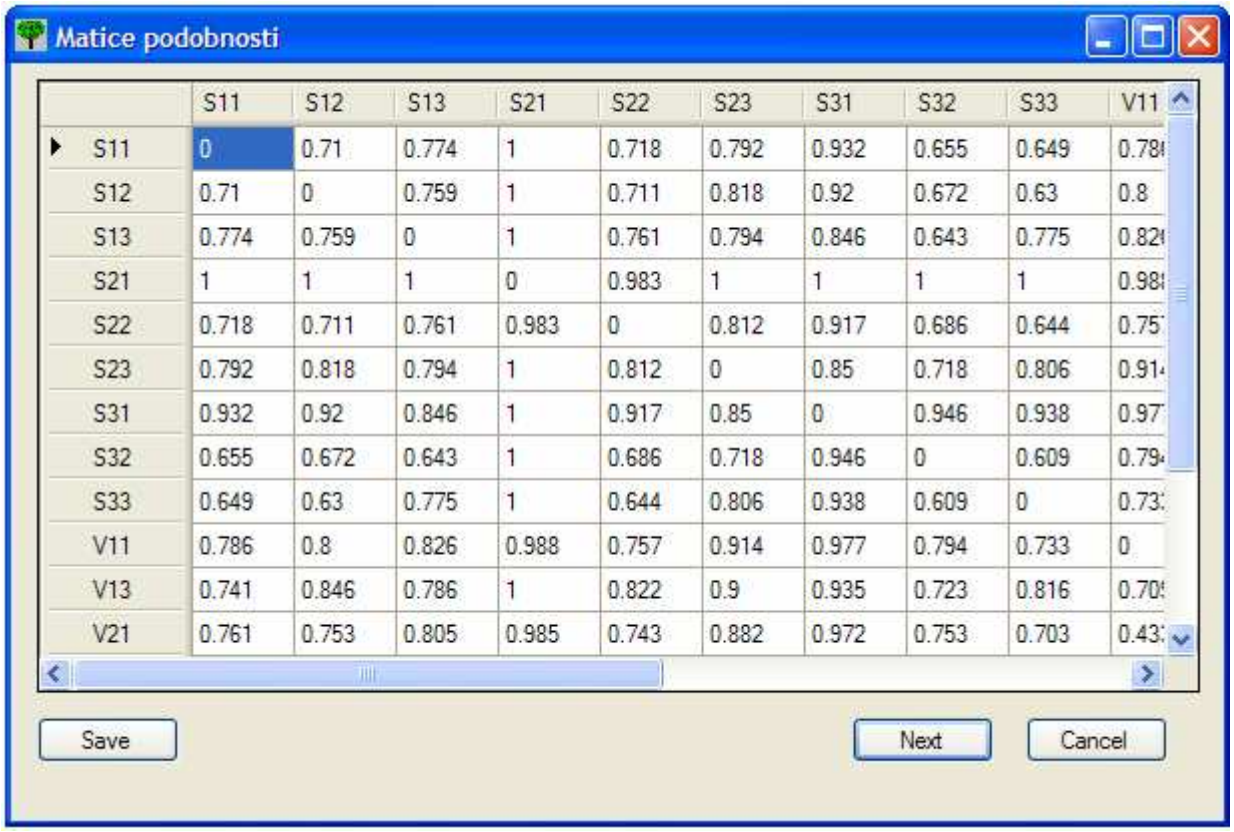


	-0-	-1-	-2-	-3-	-4-	-5-	-6-	-7-	-8-	-9-	-10-	-11-
▶	S11	0	0	55.83	0	57.66	0	0	0	0	0	0
	S12	0	52.48	55.31	0	57.15	0	0	0	0	0	0
	S13	0	0	55.83	0	0	0	0	0	0	0	0
	S21	0	0	0	0	0	0	0	60.37	0	0	0
	S22	0	52.54	0	0	57.56	0	58.75	60.42	0	0	0
	S23	0	0	0	0	0	0	0	0	60.43	0	0
	S31	0	52.58	0	0	0	0	0	0	0	0	0
	S32	0	0	0	0	57.54	0	0	0	0	0	0
	S33	0	0	55.76	0	57.58	0	0	0	0	0	62.61
	V11	0	0	55.75	0	57.18	58.09	58.81	59.78	0	61.62	62.6
	V13	0	0	55.82	0	57.15	0	58.82	0	0	61.67	62.52
	V21	51.81	0	55.79	0	57.22	0	58.85	59.92	0	0	62.75

Obr. 3.6 Upravená matice



Obr. 3.7 Matice přítomnosti a nepřítomnosti fragmentů



Obr. 3.8 Matice podobnosti