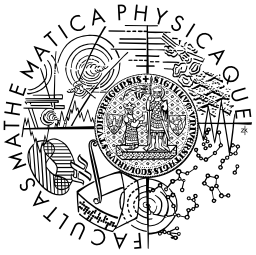


MATEMATICKO-FYZIKÁLNÍ FAKULTA
Univerzita Karlova

BAKALÁŘSKÁ PRÁCE

Ondřej Mička

Využití nekorelovaných vícebodových farmakoforových otisků při virtuálním screeningu

Katedra softwarového inženýrství

Vedoucí bakalářské práce: RNDr. David Hoksza, Ph.D.

Studijní program: Informatika

Studijní obor: Obecná informatika

Praha 2016

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně a výhradně s použitím citovaných pramenů, literatury a dalších odborných zdrojů.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona v platném znění, zejména skutečnost, že Univerzita Karlova má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle §60 odst. 1 autorského zákona.

V ..... dne ..... Podpis autora

Název práce: Využití nekorelovaných vícebodových farmakoforových otisků při virtuálním screeningu

Autor: Ondřej Mička

Katedra: Katedra softwarového inženýrství

Vedoucí bakalářské práce: RNDr. David Hoksza, Ph.D., Katedra softwarového inženýrství

Abstrakt: Nedávno byla publikována nová metoda pro virtuální screening. Tato metoda používá farmakoforové otisky a statistické metody pro vytvoření farmakoforového modelu, který je následně použit k predikci aktivity ligandů. Tato práce se zabývá dvěmi možnými vylepšeními této metody. Prvním z nich je odstranění korelovaných farmakoforů, druhé je použití větších farmakoforů (původně byly použity jen tříbodové farmakofory). Obě úpravy byly implementovány spolu s nutným rozšířením chemoinformatického softwarového balíku RDKit. Nakonec byly obě metody experimentálně vyhodnoceny a porovnány s původní metodou. Na základě těchto výsledků byla navržena a vyhodnocena další modifikace – kombinace farmakoforového modelu s podobností otisků.

Klíčová slova: virtuální screening, farmakofor, RDKit

Title: Utiliziation of uncorrelated multi-point pharmacophores in virtual screening

Author: Ondřej Mička

Department: Department of Software Engineering

Supervisor: RNDr. David Hoksza, Ph.D., Department of Software Engineering

Abstract: Recently, a new method for ligand based virtual screening was published. It uses pharmacophore fingerprints and statistical methods to create a pharmacophore model which is then used to predict the activity of ligands. In this thesis two possible enhancemets of this method were examined. The first one is the removal of correlated pharmacophores, the second one is the utilization of larger pharmacophores (originally only 3-point pharmacophores were used). Both modifications were implemented along with neccessary extension of the RDKit cheminformatics toolkit. Finally, both modifications were experimentally evaluated and compared to the original method. Based on the results, combination of the pharmacophore model with the fingerprint similarity was proposed as another modification and evaluated.

Keywords: virtual screening, pharmacophore, RDKit

Chtěl bych poděkovat vedoucímu mé práce RNDr. Davidu Hokszovi, Ph.D. za veškerou pomoc a rady, které mi při tvorbě této práce poskytl. Velké díky taktéž patří Martině Mikulů a mé rodině, kteří mne při psaní podporovali a dokázali mi zlepšit náladu, i když mi bylo nejhůř.

Zvláštní poděkování patří Katce Zákravské, Karlu Tučkovi a Ondrovi Hlavatému, kteří si i o prázdninách dokázali najít čas přečíst si prvotní verze této práce a opravit v nich chyby. Jejich připomínky a návrhy pro mne byly velmi cenným zdrojem.

Nakonec bych chtěl poděkovat za výpočetní zdroje, které byly poskytnuty Ministerstvem školství, mládeže a tělovýchovy České Repupliky v rámci projektů CESNET (projekt LM2015042) a CERIT Scientific Cloud (projekt LM2015085), spadajících do probramu Projekty velkých infrastruktur pro VaVaI.

Obsah

Úvod

High-throughput screening (HTS) je důležitou součástí procesu vývoje léčiv. Jeho cílem je identifikovat molekuly, které jsou *aktivní*, to jest, jsou schopny se vázat na cíl (např. protein) a modulovat jeho aktivitu. Během HTS je pro velké množství molekul měřena jejich schopnost se vázat na cíl.

Tento přístup má však značné nevýhody. Bývá nutné, nebo alespoň žádoucí, otestovat velké databáze obsahující klidně až desítky či stovky tisíc molekul. To může být velmi komplikované (a též drahé), dokonce i nemožné, pokud některé molekuly nejsou k dispozici, či dokonce ještě nebyly syntetizovány. Navíc, obvykle bude drtivá většina testovaných molekul neaktivní a pouze malý zlomek budou potenciálně užitečné molekuly. Je tedy snaha celý proces rozdělit – nejdříve se nějakou méně náročnou metodou zbavit „zjevně neaktivních“ molekul a až na takto zredukované databázi provést náročný HTS. A právě k tomu je používán *virtuální screening*.

Virtuální screening (VS) je počítačová metoda, jejímž cílem je určit schopnost molekuly vázat se na cíl. Jejím výsledkem je buď ohodnocení molekul v databázi podle toho, jak dobře se dokáží vázat na cíl, nebo rovnou rozdělení na aktivní a neaktivní. Na základě této informace je poté možno z databáze vybrat jen ty nejperspektivnější molekuly.

Existují dva základní přístupy k VS. První z nich je tzv. *strukturní* (structure-based virtual screening – SBVS). Při něm se snažíme nasimulovat navázání molekuly (ligandu) na cíl a následně ohodnotit stabilitu komplexu ligand-cíl. Nevýhodou tohoto přístupu je nutnost znát 3D strukturu cíle, navíc tyto simulace bývají výpočetně poměrně náročné. Výhodou ovšem je, že není třeba předem znát nějaké aktivní molekuly. Naopak druhý přístup, *ligandový* (ligand-based virtual screening – LBVS), je založen na znalosti nějakých aktivních molekul. LBVS využívá tzv. *podobnostní princip* (similar property principle), který říká, že je pravděpodobné, že podobné molekuly budou mít podobné vlastnosti. Molekuly v databázi tudíž ohodnotíme podle jejich podobnosti se známými aktivními molekulami. Jelikož metoda, kterou se tato práce zabývá, spadá pod LBVS, podíváme se na něj trochu podrobněji.

LBVS je možno rozdělit do několika kategorií podle toho, jakým způsobem se posuzuje podobnost molekul. Přehled hlavních přístupů uvádí například Taboureau a kol. (2012). Existují metody, které posuzují podobnost molekul pomocí jejich překrytí (překrývání může být založeno na elektrostatických potenciálech apod., či přímo na tvaru molekul obarveném vlastnostmi atomů). Dále existují deskriptorové metody založené na měření podobnosti molekulových otisků nebo jiných numerických popisů molekul. Pak existují farmakoforové metody, které z molekul se známou aktivitou vytvoří model popisující prostorové uspořádání vlastností, které musí aktivní molekula mít (farmakofor). Taktéž metody strojového učení mohou být použity jako LBVS. Hranice tohoto rozdělení samozřejmě nejsou ostré a různé přístupy mohou být kombinovány.

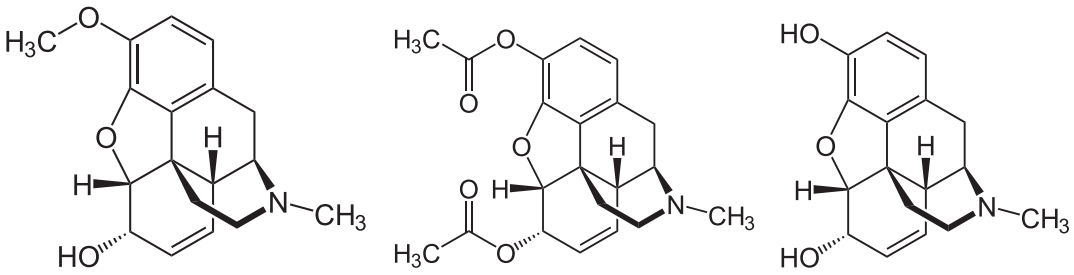
Hoksza a Škoda (2014) představili ve svém článku novou farmakoforovou metodu. Na rozdíl od obvyklých farmakoforových metod však tato metoda používá 2D farmakofory (běžné jsou 3D farmakofory), není tedy třeba znát trojrozměrnou strukturu ligandů, ale stačí znát molekulový graf[1]. V prvním kroku této metody je vytvořena reprezentace molekul pomocí farmakoforových otisků – binárních vektorů zaznamenávajících přítomnost jednotlivých farmakoforů. Následně je pomocí statistických metod vytvořen z otisků farmakoforový model. Nakonec je každá molekula v databázi ohodnocena podle její podobnosti s modelem.

Cílem této práce je prozkoumat možné zlepšení této metody. Primárně se budeme zabývat korelační analýzou a použitím vícebodových farmakoforů. Krátce se také zmíníme o možnosti kombinace farmakoforového modelu s podobností molekulových otisků. Součástí práce je též implementace zmíněných modifikací.

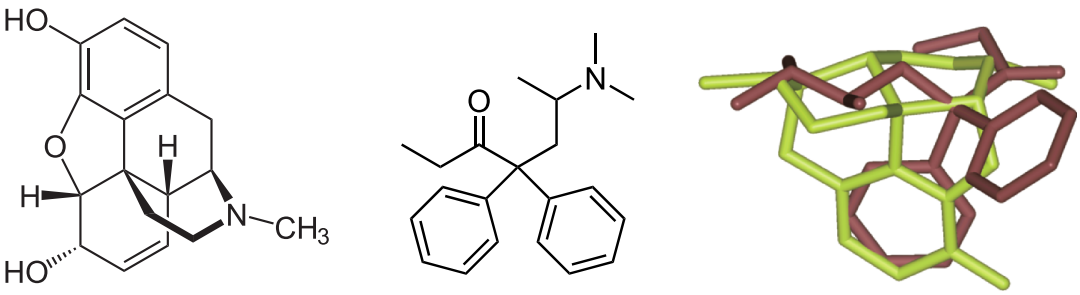
[1]Molekulový graf je jedna z běžných reprezentací molekuly. Jeho vrcholy jsou atomy a hrany odpovídají vazbám mezi nimi. Jak vrcholy, tak hrany jsou označeny typem atomu resp. vazby.

1. Virtuální screening založený na ligandech

LBVS využívá znalosti nějakých aktivních molekul k nalezení dalších. K tomu se používá tzv. *podobnostní princip* (similar property principle), který říká, že podobné molekuly mají s větší pravděpodobností podobné vlastnosti. Klasickým příkladem tohoto principu jsou morfin, heroin a kodein, ligandy opioidních receptorů (obrázek 1.1). Nicméně, velmi záleží na tom, jak podobnost molekul posuzujeme. Leach a Gilletová (Leach a Gillet, 2007, kap. 5) uvádějí jako příklad právě zmíněné ligandy opioidních receptorů. Molekulový graf metadonu, dalšího z ligandů opioidních receptorů, je mnohem méně podobný morfinu než heroin a kodein (obrázek 1.2). Pokud ale překryjeme 3D struktury morfinu a metadonu, zjistíme, že se část jejich struktury velmi dobře překrývá. Hledání spolehlivých metod měření podobnosti je proto věnováno značné úsilí.



Obrázek 1.1: Kodein, heroin a morfin, příklady ligandů opioidních receptorů.



Obrázek 1.2: Zatímco molekulové grafy morfinu (nalevo) a metadonu (uprostřed) vypadají velmi odlišně, jejich struktury mají značný překryv (morfin je zeleně, metadon hnědě). Obrázek překryvu je převzat z knihy Leache a Gilletové (Leach a Gillet, 2007, kap. 5)

Zde rozebereme dva přístupy, které se nejvíce projevují v metodě, jež představili Hoksza a Škoda (2014) – VS založený na farmakoforech (pharmacophore-based virtual screening – PBVS) a VS založený na podobnosti molekulových otisků.

1.1 VS založený na farmakoforech

Idea toho, že konkrétní biologická aktivita není závislá na tom, jak přesně molekula vypadá, ale spíš na nějaké části molekuly, je už poměrně stará. Její realizací je pojem *farmakoforu*, který představil Paul Ehrlich v roce 1909 (Ehrlich, 1909). Ehrlich definuje farmakofor jako „molekulární kostru, která nese (*phoros*) klíčové vlastnosti odpovědné za biologickou aktivitu léku (*pharmacon*)". V současnosti je pojem farmakofor definován organizací IUPAC jako „soubor sterických a elektronových vlastností, které jsou nutné k zajištění optimálních supramolekulárních interakcí se specifickým biologickým cílem a které spustí (nebo zablokují) jeho biologickou odezvu" (Wermuth a kol., 1998). Jelikož je tato definice poněkud kryptická, je kousek pod formální definicí ještě uvedeno „farmakofor může být považován za největšího společného jmenovatele souboru aktivních molekul".

Jak je vidět, farmakofory jsou (alespoň z definice) ideální prostředky pro realizaci VS. Kdybychom pro daný cíl měli odpovídající farmakofor, stačilo by pak najít v databázi molekuly, které jej obsahují – ty by měly být aktivní. Otázkou však je, jak takový farmakofor reprezentovat a jak ho vůbec získat.

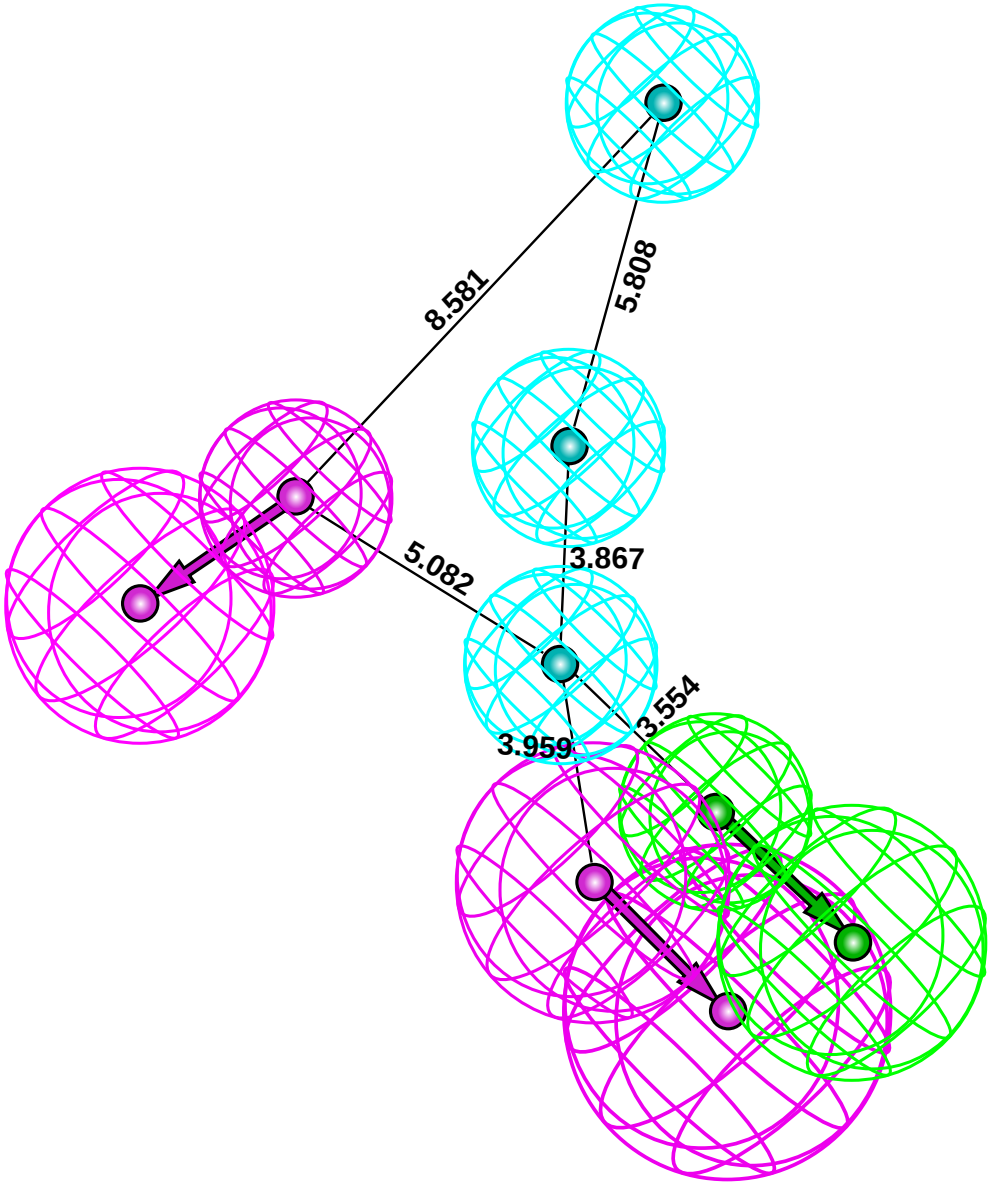
Pro praktické účely jsou předchozí definice farmakoforu poměrně abstraktní, proto bývá farmakofor často definován jako „trojrozměrné uspořádání klíčových vlastností, které umožňují molekule projevit konkrétní biologický efekt" (Dror a kol., 2004), nebo podobně (obr 1.3). Jako ony klíčové vlastnosti (tzv. *farmakoforové vlastnosti*) jsou používány převážně funkční skupiny, u kterých se předpokládá, že mají velký vliv na interakci mezi ligandem a proteinem – například akceptor či donor vodíkového můstku, nabité oblasti, aromatická jádra, ale i tzv. vyloučené objemy (oblast prostoru, do které by ligand neměl zasahovat), či roviny určující orientaci aromatických cyklů. Uspořádání těchto vlastností v prostoru nebývá dáno striktně (tj. každá vlastnost nemá přesně určený bod v prostoru, kde musí ležet), ale je dáno zónou tolerance, ve které musí vlastnost ležet (Sun, 2008).

Farmakoforový screening probíhá ve dvou fázích. V první je vytvořen *farmakoforový dotaz*, který spočívá v identifikaci farmakoforu a jeho reprezentaci. Je třeba vyextrahovat ze známých ligandů[1] farmakoforové vlastnosti a identifikovat společnou konfiguraci. Jednou z nástrah, se kterou je třeba se potýkat, je flexibilita ligandů, tedy to že existuje více konformací ligandu. Podrobněji je celý postup tvorby farmakoforového modelu popsán Drorem (Dror a kol., 2004) spolu s přehledem programů pro tvorbu farmakoforových dotazů.

Ve druhé fázi jsou potom v databázi nalezeny molekuly vyhovující farmakoforovému dotazu. K tomu jsou používány různé metody pro vyhledávání podstruktur v databázi trojrozměrných struktur molekul, například pomocí grafového isomorfismu (Brint a Willett, 1987).

Výhodou farmakoforového přístupu je abstraktnost farmakoforů. Díky tomu lze najít molekuly s podobnou biologickou aktivitou, i když mají jiný chemotyp než ligandy, z nichž jsme tvořili farmakofor, např. (Sun a kol., 2004). Jednou z nevýhod pak může být nespecificita farmakoforu, pokud ho vytváříme z různorodých ligandů.

[1] Farmakoforový dotaz je možno vytvořit i ze znalosti cíle. Tímto přístupem se zde ale zabývat nebudeme, neboť s naší prací nesouvisí.



Obrázek 1.3: Ukázka farmakoforu. Body reprezentují polohy jednotlivých vlastností (akceptor a donor vodíkového můstku a hydrofobní centra, rozlišené po řadě zelenou, fialovou a modrou), koule kolem nich zónu tolerance, šipky potom směr vodíkového můstku. Obrázek převzat (a upraven) od Kumara a Sureshe (Kumar a Suresh, 2013)

1.2 VS pomocí molekulových otisků

Mezi běžné způsoby, jak reprezentovat molekuly v počítači, patří *molekulové otisky* (fingerprints – FP). Ty vychází z toho, že často není třeba znát přesnou strukturu molekuly, ale stačí mít informaci o nějakých klíčových vlastnostech či částech molekuly.

Molekulový otisk je řetězec bitů indikující výskyt nějakých vlastností v molekule.[2] Vzhledem k tomu, že velká část otisků (speciálně všechny, které se v této práci vyskytnou) je založena na indikování přítomnosti nějakých podstruktur[3] (či fragmentů) v molekule, budeme dále mluvit jen o nich. Nicméně i jiné FP fungují velmi podobně.

Existují dva základní typy otisků – *hešované* a *strukturní* (*structural keys*). Strukturní otisky mají pevně danou množinu podstruktur (slovník fragmentů) a její mapování na bity – každý bit reprezentuje přítomnost („1“) nebo nepřítomnost („0“) dané podstruktury v molekule. Oproti tomu hešované otisky žádnou takovou předdefinovanou sadu fragmentů nemají a v principu dokáží reprezentovat libovolnou možnou podstrukturu. Daní za to je, že jednotlivé bity nemohou přímo reprezentovat podstruktury jako u strukturních otisků, ale je nutné podstruktury zahešovat a až na základě hešů nastavit odpovídající bity v otisku. Kvůli kolizím[4] nedokážeme některé podstruktury odlišit, což snižuje rozlišovací schopnosti takových otisků. Strukturní otisky tímto neduhem netrpí, ale při jejich použití je třeba vhodně zvolit množinu podstruktur.

Mezi populární FP patří *MACCS keys*, *Enhanced Connectivity Fingerprints* (ECFP), *Atom Pairs* (AP) a *Topological Torsion Fingerprints* (TT). S výjimkou MACCS keys jsou všechny z nich hešované. MACCS keys byly vyvinuty společností MDL Information Systems a obsahují dvě množiny fragmentů, menší se 166 fragmenty a větší s 960 (v nichž je zahrnuto i oněch 166 fragmentů z menší množiny) (McGregor a Pallai, 1997). V současnosti jsou implementovány (ale jen s menší množinou fragmentů) i například v RDKitu (Landrum, 2006).

Atom Pairs (Carhart a kol., 1985) jsou založené na popisu dvojic atomů a nejkratších cest mezi nimi. Pro každou dvojici atomů (krom vodíků) je vygenerován deskriptor ve tvaru:

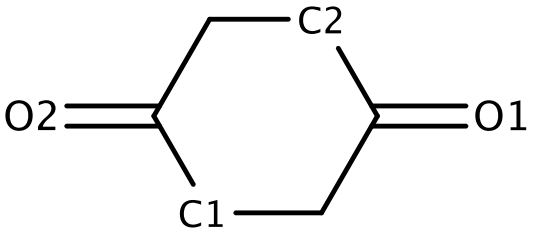
⟨ popis prvního atomu ⟩ – ⟨ vzdálenost ⟩ - ⟨ popis druhého atomu ⟩,

kde ⟨vzdálenost⟩ je počet atomů na nejkratší cestě mezi oběma atomy v molekulovém grafu. Popis atomu sestává z jeho typu (značky prvku), počtu vazeb, které z něj vychází (krom vazeb s vodíky), a počtu vazebných π-elektronů. Na obrázku 1.4 jsou ukázky deskriptorů.

[2] Existují i varianty FP které místo pouhé přítomnosti vlastností reprezentují i jejich počet (má-li to smysl, např. pokud jde o přítomnost nějaké podstruktury v molekule), nicméně tyto dva přístupy se od sebe téměř neliší. Dokonce většina metod pro tvorby FP dokáže fungovat v obou variantách.

[3] Za podstrukturu budeme považovat v podstatě libovolnou množinu atomů s nějakými vztahy mezi nimi. Podstrukturou může být například karbonylová skupina, atom uhlíku sousedící se dvěma jinými atomy než je vodík, dvě aromatická jádra vzdálená od sebe 2 Å, ...

[4] Kolize nastává, pokud různé podstruktury mají stejný heš, a tedy nastaví v otisku stejné bity.



Obrázek 1.4: Ukázka Atom Pairs. Cyklohexa-1,4-dion má osm atomů a tedy celkem 28 dvojic atomů. Některé jsou ale identické (mají stejný deskriptor), různých dvojic je jen jedenáct. Jak dvojici atomů C1, O1, tak dvojici C2, O2 odpovídá stejný deskriptor – `CX2-(4)-O·X1`. Tečka („·“) značí přítomnost jednoho π-elektronu, Xn pak vyjadřuje n vazeb vycházejících z atomu (nepočítaje vazby s vodíky). Další z deskriptorů jsou například `CX2-(4)-CX2` (pro dvojici atomů C1 a C2) a `O·X1-(6)-O·X1` (pro dvojici atomů O1 a O2).

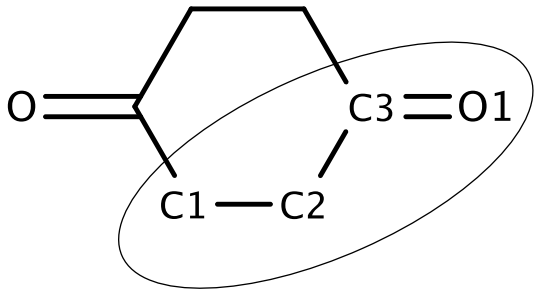
Topological Torsion (Nilakantan a kol., 1987) využívá lineární čtyřatomové fragmenty. Pro každou čtveřici lineárně spojených atomů (vyjma vodíků) je vytvořen deskriptor:

⟨popis atomu 1⟩ – ⟨popis atomu 2⟩ – ⟨popis atomu 3⟩ – ⟨popis atomu 4⟩,

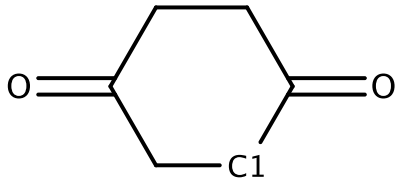
přičemž atomy jsou v pořadí, v jakém jsou spojeny vazbami. Popis atomu opět sestává z počtu vazebných π-elektronů, typu atomu (značky prvku) a počtu vazeb, které z atomu vychází (krom vazeb s vodíky a vazeb, které jsou součástí fragmentu). Ukázka deskriptoru je na obrázku 1.5.

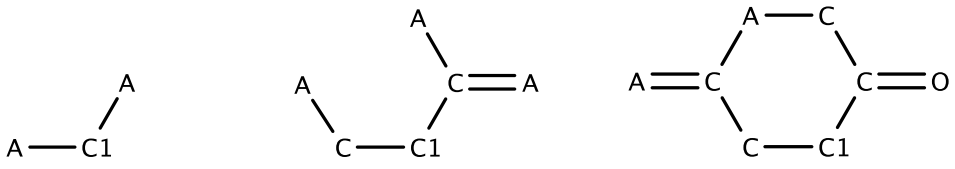
Nejnovější a nejkomplikovanější z těchto čtyř jsou ECFP (Rogers a Hahn, 2010). Ty vychází z Morganova algoritmu (Morgan, 1965), který je používán pro určení isomorfismu dvou molekulových grafů. ECFP iterativně buduje deskriptory jednotlivých atomů. V nulté iteraci je každému atomu (krom vodíků) přiřazen deskriptor, který je závislý jen na typu atomu (a jeho vlastnostech). V každé další iteraci je deskriptor atomu upraven na základě deskriptorů jeho sousedů (tj. atomů, se kterými je spojen vazbou), přičemž tento deskriptor je přidán do otisku. V i-té iteraci tudíž deskriptor popisuje okolí atomu až do vzdálenosti i vazeb (viz obrázek 1.6). Podle průměru největšího okolí, který je roven polovině počtu iterací, jsou rozlišovány ECFP4 (poloměr čtyři, tedy dvě iterace) a ECFP6 (poloměr šest, tři iterace), případně další.

Velkou výhodou molekulových otisků je, že jde o velmi efektivní reprezentaci molekuly v počítači – práce s bitovými řetězci je mnohem rychlejší než například práce s grafy. Nevýhodou ovšem je, že FP uchovávají jen část informací o molekule, například není možné z otisku zjistit, jakou molekulu reprezentuje. To ale často nevadí – například při hledání podstruktur jsou pomocí FP z databáze vyfiltrovány molekuly, které danou podstrukturu *určitě nemají*, což snadno poznáme podle toho, zda má otisk molekuly nastavené všechny bity, které má nastavené otisk podstruktury. Dalším případem je hledání podobných molekul. Molekuly se stejnými otisky se sice mohou lišit, ale molekuly s „rozdílnými“ otisky se většinou liší mnohem více.



Obrázek 1.5: Vyznačené atomy C1, C2, C3 a O1 tvoří lineární čtyřatomový fragment. Odpovídající deskriptor pro TT je `(0-C-1)-(0-C-0)-(1-C-1)-(1-O-0)`. Závorky postupně reprezentují atomy C1, C2, C3 a O1, první číslo v závorce je počet π-elektronů na atomu, následuje značka prvku a počet vazeb, které z atomu vycházejí a nejsou součástí fragmentu (opět krom vazeb s vodíky).





0. iterace 1.iterace 2.iterace

Obrázek 1.6: Přiřazování deskriptorů v ECFP pro molekulu cyklohexa-1,4-dionu. V nulté iteraci obsahuje deskriptor přiřazený atomu C1 pouze informace o samotném atomu. Po první iteraci obsahuje deskriptor i informace o sousedech C1 a ve druhé iteraci už i o sousedech sousedů C1. Ve třetí iteraci už by deskriptor obsahoval informace o celé molekule.

Pro určení míry podobnosti dvou otisků se používají takzvané *podobnostní koeficienty*. Asi nejpopulárnějším podobnostním koeficientem je *Tanimotův koeficient* (lze se s ním setkat i pod názvem *Jaccardův koeficient*). Ten určuje podobnost otisků jako

$$\frac{c}{a+b-c,}$$

kde a resp. b jsou počty bitů nastavených na jedničku prvního resp. druhého otisku, c je počet bitů nastavených na jedničku jak v prvním, tak ve druhém otisku. Kromě Tanimotova koeficientu existují ještě i další, například kosinový koeficient, Russel-Raoův koeficient či Hammingova vzdálenost (která, jak už název napovídá, určuje na rozdíl od předchozích míru odlišnosti dvou otisků). Podrobněji se podobnostním koeficientům věnuje například Willett (2006).

Ve virtuální screeningu se molekulové otisky využívají právě k hledání podobných molekul (Willett, 2011), které by podle podobnostního principu měly mít podobnou biologickou aktivitu. Molekuly v databázi dostanou skóre podle podobnosti jejich otisků s otiskem známé aktivní molekuly. Často bývá známo více aktivních molekul, v takovém případě je možné zkombinovat podobnosti vůči všem známým molekulám pomocí *data fusion*. Nejjednoduššími metodami data fusion jsou *max-fusion* a *mean-fusion*, kdy skóre molekuly $S(M)$ je určeno jako

$$S(M) = \max_{A \in \mathcal{A}} s(M,A),$$

resp.

$$S(M) = \frac{1}{|\mathcal{A}|} \sum_{A \in \mathcal{A}} s(M,A),$$

kde $\mathcal{A}$ jsou známé aktivní molekuly a $s(M,A)$ je podobnostní koeficient otisků molekul M a A.

2. 2D farmakoforová metoda

V této kapitole podrobně popíšeme metodu, kterou Hoksza a Škoda navrhli ve svém článku *2D Pharmacophore Query Generation* (Hoksza a Škoda, 2014) (dále jen 2DPP metoda). Ta je založena na použití 2D farmakoforového modelu. Také zde představíme modifikace této metody, kterými jsme se zabývali.

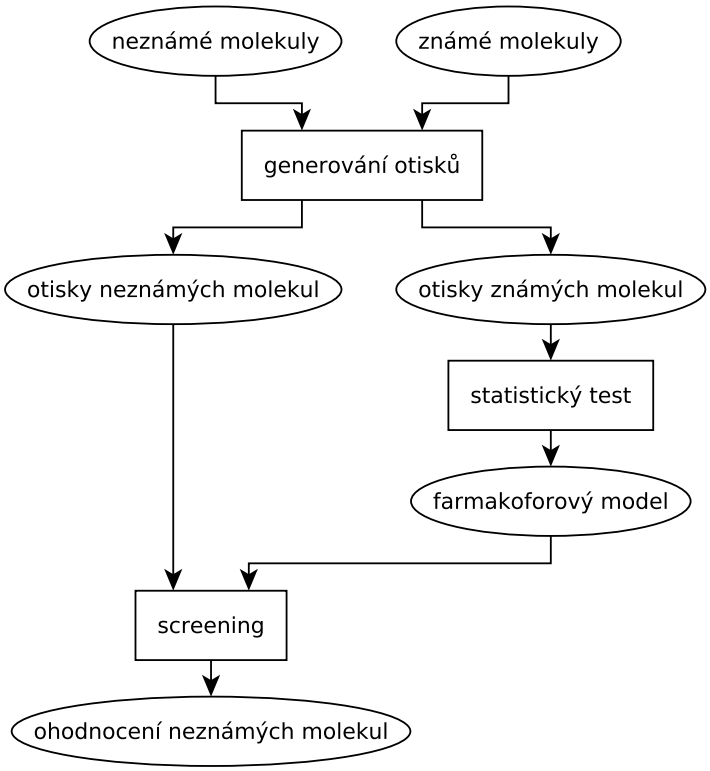
2DPP metoda funguje podobným způsobem jako většina farmakoforových metod – na základě znalosti několika aktivních a (což není běžné) neaktivních molekul určí vlastnosti, které je odlišují, z těch následně sestaví farmakoforový model, pomocí kterého je pak proveden samotný screening. Zásadní rozdíl oproti běžným farmakoforovým metodám je v tom, že používá *topologické (2D) farmakofory*. Ty se od běžných (3D) farmakoforů liší tím, že se nejedná o uspořádání farmakoforových vlastností v prostoru, ale v molekulovém grafu, tj. vzdálenosti v trojrozměrném prostoru jsou nahrazeny vzdálenostmi v molekulovém grafu[1]. Samozřejmě je mírně odlišná i množina vlastností, neboť některé nemají ekvivalent v molekulovém grafu (například vyloučené objemy).

Výhodou 2D farmakoforů oproti těm 3D je, že se můžeme vyhnout problémům, které vyvstávají při práci s 3D strukturou molekuly, například není nutné řešit, že má molekula více konformací. Nicméně cenou za to je další potenciální snížení přesnosti – 2D farmakofory vychází z předpokladu, že topologické vzdálenosti atomů odpovídají jejich vzdálenosti v prostoru. To je sice často pravda, ale ne vždy. Není těžké najít molekulu obsahující dva atomy, které jsou od sebe v molekulovém grafu daleko, ale ve skutečnosti jsou si blízko (např. různé proteiny nebo RNA).

Dalším specifikem 2D farmakoforové metody je, že kombinuje farmakoforový přístup s otisky použitím *farmakoforových otisků*. Myšlenkou za farmakoforovými otisky je definovat nějakou množinu farmakoforů (farmakoforový prostor) a molekuly reprezentovat pomocí otisků, které reflektují přítomnost těchto farmakoforů (jde tedy o strukturní otisky). Vzhledem k tomu, že farmakoforů (tak, jak jsme je popsali v předchozí kapitole) je nekonečně mnoho, je třeba uvažované farmakofory omezit. Prvním omezením je limit na počet vlastností ve farmakoforech, které tak například mohou obsahovat jen tři vlastnosti. Druhým omezením je diskretizace spojitých vzdáleností mezi vlastnostmi. To může být provedeno například definováním konečné sady diskrétních vzdáleností – „přihrádek“ (či košů z anglického *bins*), každé přihrádce pak odpovídá nějaký interval vzdáleností v prostoru (Dror a kol., 2004).

Metoda sestává ze tří základních částí (viz obrázek 2.1). První je vytvoření farmakoforových otisků jak známých aktivních a neaktivních molekul, tak molekul z databáze, kterou chceme prozkoumat. Druhou částí je nalezení signifikantních farmakoforů a vytvoření farmakoforového modelu. Poslední částí je pak samotný screening databáze pomocí modelu. V následujících odstavcích tyto části rozebereme podrobně.

[1] Vzdálenost dvou atomů v molekulovém grafu odpovídá nejmenšímu počtu vazeb, které je oddělují, tedy krajní molekuly uhlíku v propanu jsou od sebe vzdáleny 2.



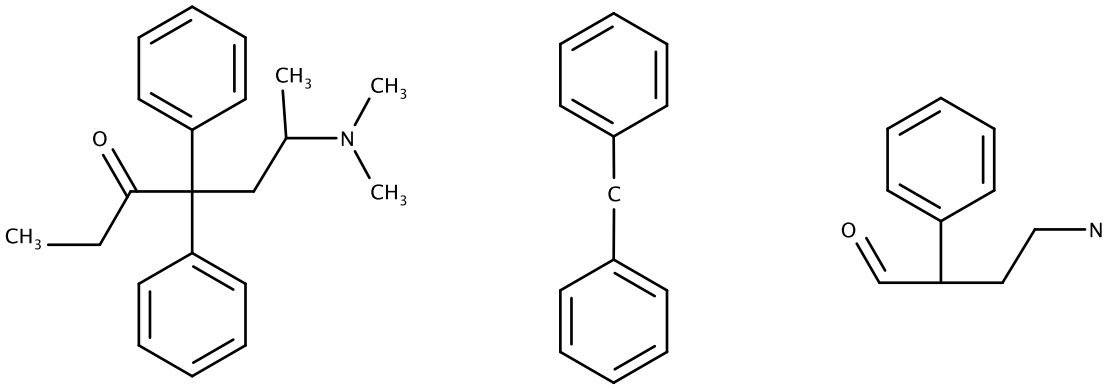
Obrázek 2.1: Schéma průběhu 2D farmakoforové metody.

2.1 Tvorba farmakoforových otisků

Ke tvorbě farmakoforových otisků pro tuto metodu byl použit RDKit (Landrum, 2006). RDKit je svobodný softwarový balík a stejnojmenná knihovna pro jazyk Python, která obsahuje širokou škálu nástrojů pro chemoinformatiku. Součástí RDKitu je i modul `Chem.Pharm2D`, který obsahuje funkcionalitu pro tvorbu 2D (topologických) farmakoforových otisků.

Topologický N-bodový farmakofor, tak jak je naimplementován ve výše zmíněném modulu, je N-tice (ne nutně různých) farmakoforových vlastností spolu s jejich „zapřihrádkovanými“ topologickými vzdálenostmi (obrázek 2.2). Přihrádkování vzdáleností funguje stejně jako diskretizace vzdáleností u 3D farmakoforových otisků – každé přihrádce odpovídá nějaký interval topologických vzdáleností. U topologických farmakoforů, na rozdíl od jejich 3D protějšků, není přihrádkování nutné (můžeme použít rovnou topologické vzdálenosti), ale umožňuje snížit velikost farmakoforového prostoru a dává vzdálenostem vlastností jistou míru tolerance, podobně jako u 3D farmakoforů.

Autoři použili v metodě dvou- a tříbodové farmakofory, tj. otisky obsahovaly oba typy farmakoforů. RDKit v současné verzi (2015.09.1) implementuje pouze dvou- a tříbodové farmakofory. Jelikož žádný jiný svobodný softwarový balík tento typ farmakoforů neimplementuje, bylo jedním z cílů této práce rozšířit implementaci RDKitu na obecné N-bodové farmakofory, což nám umožnilo použít vícebodové farmakofory v 2DPP metodě.



Obrázek 2.2: Molekula metadonu (nalevo) a dva topologické farmakofory, které obsahuje. Uprostřed je dvoubodový farmakofor, tvořený dvěma aromatickými jádry s (topologickou) vzdáleností dvě. Napravo je tříbodový farmakofor tvořený aromatickým jádrem a dvěma donory vodíkového můstku (kyslík a dusík), vzdálenost kyslík – dusík je pět, kyslík – arom. jádro je tři a dusík – arom. jádro je čtyři. Při zapřihrádkování do košů $\langle 0,3)$,$\langle 3,5)$ a $\langle 5,8)$ budou vzdálenosti v prostředním farmakoforu nula, v levém dva, jedna a jedna.

Co se týče ostatních parametrů otisků, Hoksza a Škoda použili pět farmakoforových vlastností – donor a akceptor vodíkového můstku, aromatické jádro, kladně ionizovatelný atom a záporně ionizovatelný atom. Přesné definice těchto vlastností jsou součástí Přílohy 3, použitý formát je definován v dokumentaci RDKitu (Landrum, 2015a, sekce 8.3). Vzdálenosti vlastností jsou přihrádkovány podle tří intervalů (košů) – $\langle 0,2)$, $\langle 2,5)$ a $\langle 5,8)$. Speciálně tedy farmakofory, kde by byly dva atomy od sebe vzdálené osm a více vazeb, se vůbec neuvažují.

2.2 Farmakoforový model

Vytvořením farmakoforových otisků jsou v každé molekule identifikovány přítomné farmakofory. Ve druhé fázi 2DPP metody jsou tyto informace využity k nalezení farmakoforů, které mají vliv na aktivitu ligandů vůči cíli, a sestavení *farmakoforového modelu*. Farmakoforový model není nic jiného než farmakoforový otisk fiktivní molekuly (a je tedy reprezentován jako bitový vektor), která obsahuje právě ty farmakofory, které odlišují aktivní molekuly od těch neaktivních. Zde je třeba podotknout, že jsou uvažovány pouze farmakofory, které *způsobují* aktivitu, ne farmakofory, které naopak aktivitě zabraňují (tj. farmakofory, které se vyskytují v neaktivních molekulách, ale ne v aktivních). Předpokládá se, že pravděpodobnost výskytu farmakoforů zabraňujících aktivitě je zanedbatelná.

Identifikace farmakoforů, jež odlišují aktivní molekuly od neaktivních (takzvané *signifikantní farmakofory*), je provedena pomocí statistického testu. Pro každý farmakofor je testována hypotéza, zda se výskyt tohoto farmakoforu v aktivních a neaktivních molekulách signifikantně liší. Nulová hypotéza je, že se procento aktivních molekul obsahujících daný farmakofor významně neliší od procenta neaktivních molekul obsahujících onen farmakofor. Alternativní hypotéza je, že se procento výskytů liší a tudíž je farmakofor signifikantní. Hladina testu byla parametrem, který autoři zkoumali.

K otestování této hypotézy se nabízí dva testy – dvouvýběrový z-test (pro rovnost poměrů) a Fisherův test. Podmínky na nezávislost jsou splněny (molekuly jsou navzájem nezávislé), nicméně z-test vyžaduje, aby data pocházela z normálního rozdělení, nebo alespoň aby byl vzorek dostatečně velký (deset či více pozorování každé skupiny). To je ovšem předpoklad, který může být obtížné splnit – nemáme zaručeno, že budeme mít dostatečné množství známých aktivních molekul. Z toho důvodu se autoři rozhodli pro použití Fisherova testu, který není omezen touto podmínkou. Konkrétně použili implementaci Fisherova testu obsaženou v jazyce R (Ihaka a Gentleman, 1996)[2]

Použití statistického testu k vytvoření modelu vyžaduje, abychom znali krom aktivních molekul i nějaké neaktivní molekuly, neboť je nutné znát podíl molekul obsahujících daný farmakofor jak mezi aktivními, tak mezi neaktivními molekulami. V případě, že budeme znát pouze aktivní molekuly, autoři navrhují náhodně vybrat molekuly z velké veřejně přístupné databáze a ty použít jako známé neaktivní molekuly. Tento krok ospravedlňují tím, že aktivních molekul bude v takové databázi velmi málo, a tudíž pravděpodobnost, že při náhodném výběru narazíme zrovna na ně, bude velmi malá.

2.3 Screening

Máme-li farmakoforový model a otisky molekul v databázi, můžeme začít se samotným screeningem. Běžnou praxí u farmakoforových metod je, že screening probíhá formou odfiltrování molekul nevyhovujících modelu. Topologické farmakofory jsou však pouze aproximací trojrozměrných farmakoforů, a proto se autoři rozhodli místo filtrování ohodnotit molekuly v databázi pomocí skóre, které odráží, jak moc molekula „vyhovuje“ modelu. Tomuto přístupu nahrává to, že i model je reprezentován jako farmakoforový otisk, takže je možné využít podobné metody jako při práci s molekulovými otisky. Není ale úplně vhodné použít například Tanimotův koeficient, neboť ten vyjadřuje podobnost otisků. Molekula, obsahující všechny farmakofory obsažené v modelu, by tak mohla dostat malé skóre jen proto, že obsahuje i velké množství jiných farmakoforů, což není žádoucí (model říká pouze jaké farmakofory „podporují“ aktivitu, o ostatních neříká nic).

Nejjednodušší způsob ohodnocení, který se nabízí je počet farmakoforů, které má molekula společné s modelem (tj. počet společných bitů v otisku). Jak autoři ale uvádějí, toto ohodnocení trpí zásadním nedostatkem – různých hodnot skóre je jen tolik, kolik je signifikantních farmakoforů (tj. počet bitů nastavených na jedna v modelu). Pokud tedy model obsahuje jen 30 signifikantních farmakoforů (řádově tolik jich obsahovaly modely z dat analyzovaných ve článku), pak možné hodnoty skóre jsou 0, 1, ..., 30. Takové ohodnocení může být moc hrubé, pokud je databáze velká (desítky tisíc molekul).

Autoři se proto rozhodli přidat každému farmakoforu-bitu váhu – p-hodnotu, jíž bylo dosaženo při tvorbě modelu. Krom zjemnění hodnocení oproti počítání společných bitů je tím dosaženo toho, že farmakofory, které s větší jistotu oddělují aktivní molekuly od neaktivních, mají větší váhu. Přesně je tedy skóre

[2] Z technických důvodů jsme v naší implementaci použili implementaci Fisherova testu obsaženou v knihovně Scipy (Jones a kol., 2001).

molekuly M vůči farmakoforovému modelu P určeno následovně:

$$S_{\text{pp}}(M,P) = \sum_{i=1}^{l} M[i] \cdot P[i] \cdot w(i),$$

kde l je počet bitů farmakoforových otisků, $M[i]$, resp. $P[i]$, je i-tý bit otisku molekuly, resp. modelu. Váha i-tého bitu $w(i)$ je definována jako

$$w(i) = 1 - \text{pval}[i] \cdot \frac{0{,}5}{1-\alpha},$$

kde α je hladina testu použitá při tvorbě modelu a pval$[i]$ je p-hodnota pro i-tý farmakofor-bit. Účelem váhové funkce je zobrazit p-hodnoty na interval $\langle 0{,}5; 1\rangle$, neboť ony samotné jsou malé (například při $\alpha = 0{,}05$ jsou p-hodnoty signifikantních farmakoforů menší než 0,05).

I přesto je problematické, pokud model obsahuje jen malé množství farmakoforů (nebo dokonce žádný). Proto autoři navrhli zkombinovat „farmakoforové“ skóre s ohodnocením pomocí jiné metody. Konkrétně použili max-fusion pro MACCS keys a Tanimotův koeficient. Formálně je tedy skóre spočítáno jako

$$S_{\text{comb}}(M,P) = S_{\text{pp}}(M,P) \cdot \max_{A \in \mathcal{A}} \left(T_{\text{MACCS}}(M,A) \right),$$

kde $\mathcal{A}$ je množina všech známých aktivních molekul a $T_{\text{MACCS}}(M,A)$ je Tanimotův koeficient pro MACCS keys otisky molekul A a M.

2.4 Modifikace metody

Cílem této práce je prozkoumat některé modifikace výše popsané metody, které by mohli přinést její zpřesnění. První modifikace je spíše jen prozkoumání vlivu dalšího parametru – počtu bodů ve farmakoforech. Čím více vlastností farmakofor obsahuje, tím více informace nese, je tedy přirozené se domnívat, že zvýšení počtu vlastností ve farmakoforech zvýší výkon celé metody. Na druhou stranu, pokud budou farmakofory obsahovat moc bodů, mohou být moc specifické. Doženeme-li úvahu ad absurdum, pak pro dostatečně velký počet bodů budou farmakofory prakticky odpovídat celým molekulám. Tomu se ovšem dá (alespoň částečně) vyhnout tím, že nepoužijeme jen N-bodové farmakofory, ale dvou- až N-bodové farmakofory, tj. zahrneme všechny velikosti farmakoforů mezi dvěma a N. Větším problémem však může být výpočetní náročnost. Množství možných farmakoforů roste alespoň exponenciálně s N, může se tedy snadno stát, že množství výpočetního času spotřebovaného na tvorbu otisků bude neúměrně vysoké oproti zlepšení, kterého tím dosáhneme.

Druhá modifikace vychází z myšlenky, že výskyty farmakoforů v molekulách nemusí být nezávislé. Předpokládejme, že výskyt farmakoforu A v molekule znamená, že se v molekule vyskytuje i farmakofor B a naopak. Potom v podstatě A i B odpovídají jedinému farmakoforu AB, který ale reprezentujeme dvěma způsoby. To způsobuje nežádoucí vážení – farmakofor AB bude mít při počítání skóre dvojnásobnou váhu, neboť ho jednou započítáme za farmakofor A a jednou za farmakofor B (oba se budou v modelu vyskytovat společně).

Chtěli bychom tedy takové závislosti odstranit (odstraněním některých farmakoforů), ale zároveň tím nechceme přijít o informace o žádném farmakoforu. Provedeme proto korelační analýzou – zjistíme, které farmakofory (resp. jejich výskyty) spolu korelují, a poté některé z nich odstraníme tak, aby každý odstraněný farmakofor koreloval s nějakým jiným, který jsme neodstranili.

Třetí modifikace spočívá ve změně určování skóre. Ukázalo se totiž, že tato metoda si vede dobře na datech, na kterých mají jiné otiskové metody (jako ECFP či MACCS keys) špatný výkon, ale na datech, kde ostatní metody mají dobrý výkon, si vede špatně. Proto jsme se pokusili zkombinovat ohodnocení farmakoforovým modelem s ohodnocením pomocí podobnosti se známými aktivními molekulami.

3. Vícebodové farmakoforové otisky v RDKitu

Kvalitní software je klíčovou součástí každého chemoinformatického projektu. Existuje proto celá řada jak proprietárních, tak svobodných softwarových balíků zaměřených na použití v chemoinformatice. Mezi jeden z nejpopulárnějších patří RDKit (Landrum, 2006). RDKit je svobodný softwarový balík, který poskytuje širokou paletu nástrojů pro chemoinformatiku, a to především formou knihovny pro jazyk Python, ve kterém je společně s C++ naprogramován. Jeho popularitu ilustruje například existence wrapperů, které umožňují RDKit používat v Javě či C#, i přesto, že například pro Javu existují jiné svobodné balíky (jako CDK (Steinbeck a kol., 2003)).

RDKit byl původně vyvíjen společností Rational Discovery LLC, nicméně v květnu 2006 byl uvolněn jako open-source pod BSD licencí (Landrum, 2015a, sekce 1.1.3). Jádro RDKitu je napsané v C++, nicméně poskytuje API pro Python, ve kterém je také naprogramována část funkcionality a ze kterého je RDKit nejčastěji používán. Mezi nástroje, které RDKit poskytuje, patří hledání podstruktur, práce s chemickými reakcemi, různé otisky (např. AP, TT, MACCS keys, ...), (de)serializace molekul s podporou různých formátů jako například SMILES či SDF, různé deskriptory, práce s 3D strukturami a farmakofory, ale i různé metody strojového učení. Nás ovšem budou zajímat 2D farmakoforové otisky.

V této kapitole popíšeme stávají implementaci 2D farmakoforových otisků v RDKitu a změny v ní, které umožnily použití obecných N-bodových farmakoforů oproti nejvýše tříbodovým, které jsou v současnosti podporovány. Dále představíme drobnou modifikaci, která celé generování otisků zrychlila a umožnila tak použít až pětibodové farmakofory.

Verze RDKitu, kterou jsme upravovali a kterou zde tudíž budeme používat, je 2015.09.1 (verze aktuální v době počátku této práce). Vzhledem k tomu, že nové stabilní verze jsou vydávány každý půlrok, je v současné době aktuální už novější verze, 2016.03.1. V této verzi však nedošlo k žádným změnám v částech, které s touto prací souvisí, proto by neměl být problém na ni aplikovat naše změny. Upravené zdrojové kódy implementující naše modifikace se nacházejí v Příloze 2.

3.1 Modul `Chem.Pharm2D`

Implementace 2D farmakoforových otisků a funkcionality s nimi související se nachází v modulu `Chem.Pharm2D`. Hlavními částmi tohoto modulu je třída `SigFactory` (obsažená ve stejnojmenném modulu), která spravuje parametry otisku (který je též označován jako *signature*) a funkce `Gen2DFingerprint` (obsažená v modulu `Generate`), která vytvoří samotný otisk. Dále modul ještě obsahuje přednastavenou `SigFactory` s parametry z článku Gobbiho a Poppingera (Gobbi a Poppinger, 1998) a nedokončený líný generátor otisků.

Použití modulu je poměrně snadné. Nejdříve je třeba definovat vlastnosti, které budou ve farmakoforech použity, a načíst je. Následně je nutné vytvořit instanci `SigFactory` a inicializovat ji žádanými parametry – vlastnostmi, minimálním a maximálním počtem bodů ve farmakoforech a množinou košů pro přihrádkování vzdáleností. Máme-li správně nastavenou instanci `SigFactory`, můžeme ji spolu s molekulou předat jako parametr funkci `Gen2DFingerprint`, která (jak už název napovídá) vytvoří samotný otisk. Otisk je (řídký) bitový vektor, uchovávající pouze to, které bity otisku jsou nastavené na jedničku. Informace o parametrech či o tom, který bit reprezentuje který farmakofor, je možno zjistit z instance `SigFactory`.

Konkrétní použití v programu může vypadat následovně. Nejdříve ze souboru `featFile.fdef` načteme definice vlastností:

```
>>> from rdkit.Chem import ChemicalFeatures
>>> featFactory = ChemicalFeatures.BuildFeatureFactory('featFile.fdef')
```

Poté vytvoříme instanci `SigFactory`:

```
>>> from rdkit.Chem.Pharm2D.SigFactory import SigFactory
>>> sf = SigFactory(featFactory,minPointCount=2,maxPointCount=3)
>>> sf.SetBins([(0,2),(2,5),(5,8)])
>>> sf.Init()
```

Nyní je všechno připraveno pro vygenerování otisků:

```
>>> from rdkit import Chem
>>> from rdkit.Chem.Pharm2D import Generate
>>> mol = Chem.MolFromSmiles('COc(c1)cccc1C#N')
>>> fp = Generate.Gen2DFingerprint(mol,sf)
```

S vytvořeným otiskem pak můžeme pracovat jako s obyčejným bitovým vektorem.

Po stránce implementace je modul mnohem méně přívětivý. Celý modul je naimplementován v jazyce Python, není to tedy jen obal nad implementací v jazyce C++. Jak již varuje poznámka z přehledu funkcionality v dokumentaci RDKitu (Landrum, 2015a, kap. 3), implementace 2D farmakoforových otisků sice je funkční, ale není kompletní, natož optimální. Dle historie tohoto modulu nalezené na GitHubu RDKitu (Landrum, 2015b) byla poslední větší změna provedena kolem roku 2010 i přesto, že z kódu to vypadá, že měla být přidána ještě další funkcionalita (viz nedokončený líný generátor otisků).

K modulu navíc prakticky neexistuje použitelná programátorská dokumentace. Dokumentace RDKitu se o 2D farmakoforových otiscích zmiňuje trochu podrobněji pouze na dvou místech – v kapitole 7.10, která obsahuje návod, jak s nimi pracovat, a v kapitole 8.4, která popisuje formát otisků, a to ještě jen prakticky pomocí jediného obrázku (obrázek 3.2). Jinak je k dispozici pouze dokumentace generovaná z dokumentačních komentářů a samotný kód. Obojí je ovšem poněkud strohé, a tak není úplně jednoduché se v kódu zorientovat. Funkce sice jsou většinou okomentované, ale ne vždy dostatečně jasně a často chybí informace o tom, co se předpokládá o argumentech. Z toho důvodu jsme se rozhodli své zásahy do kódu omezit na nutné minimum tak, aby byl kód zpětně kompatibilní. Opačný přístup by totiž pravděpodobně vyžadoval přepsat velkou část celého modulu, což nebyl cíl této práce.

3.2 Formát farmakoforového otisku

Nejdříve ze všeho musíme ukázat, jak přesně jsou farmakofory v otisku uspořádány (resp. které bity odpovídají kterým farmakoforům) a jak přesně jsou reprezentovány farmakofory. Jak se ukázalo, právě přesný formát otisků je důvodem, proč nejsou podporovány více než tříbodové farmakofory.

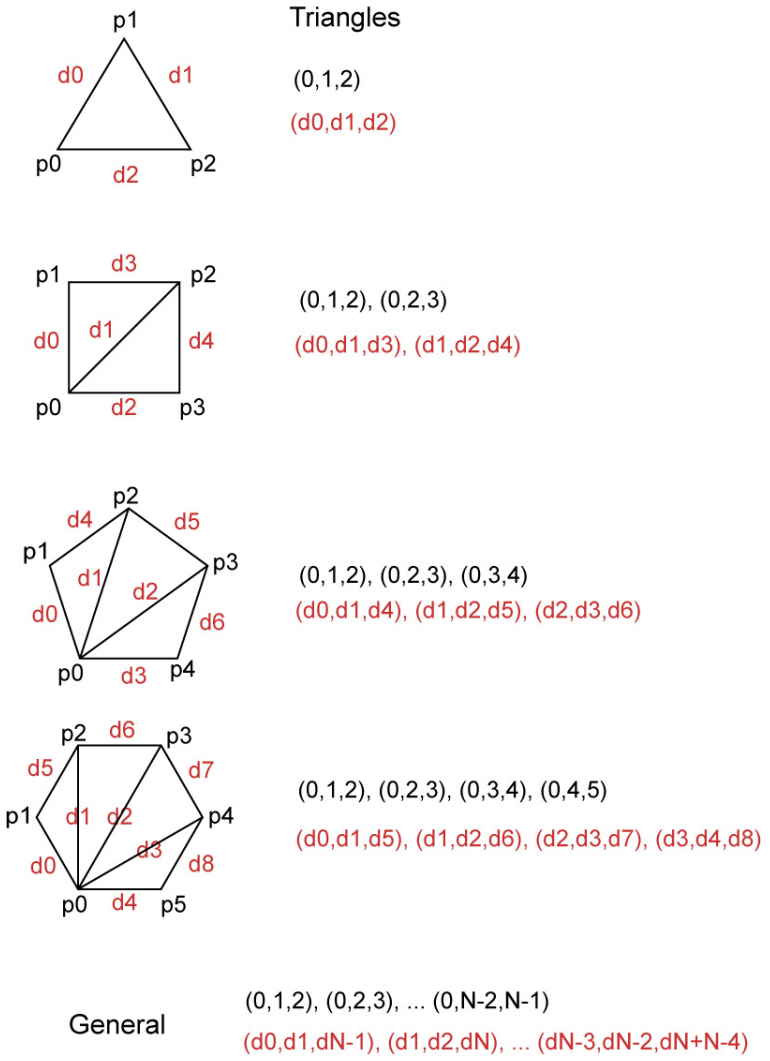
Bohužel, jako celý modul `Chem.Pharm2D`, i formát otisků je zdokumentován poměrně málo. V dokumentaci RDKitu (Landrum, 2015a) je formátu otisků věnována celá kapitola 8.4, která obsahuje pouze obrázek s ukázkou (viz obr. 3.2) a informaci o tom, že se nejedná o hešovaný otisk, ale každý bit reprezentuje jeden farmakofor. Další informací je obrázek triangulace farmakoforu, na který odkazuje zdrojový kód modulu `Chem.Pharm2D.Generate` (soubor `Generate.py` v modulu `Chem.Pharm2D`). Poněkud nečekaným zdrojem informací se ukázal být článek Gobbiho a Poppingera (Gobbi a Poppinger, 1998), ve kterém jsou použity tříbodové farmakoforové otisky založené na stejném principu jako ty z RDKitu. Posledním zdrojem jsou pak samotné zdrojové kódy modulu `Chem.Pharm2D`.

Jak jsme již řekli, N-bodový farmakofor je soubor N (ne nutně různých) farmakoforových vlastností spolu s jejich vzdálenostmi. Aby bylo zřejmé, kdy mluvíme o vlastnostech jako o obecných entitách, „nálepkách“ pro atomy (či skupiny atomů), a kdy o nich mluvíme jako o „instancích“ těchto entit ve farmakoforu, budeme výskytům vlastností ve farmakoforu říkat *body* a „nálepkám“ pouze *vlastnosti*. Každý bod má pak jednoznačně určeno, která vlastnost-nálepka mu přísluší. Dále kombinace N bodů, ale bez vzdáleností mezi nimi, je označována jako N-bodový *proto-farmakofor*, naopak kombinace vzdáleností N bodů, bez přidělených konkrétních bodů, je N-bodové *lešení* (scaffold).

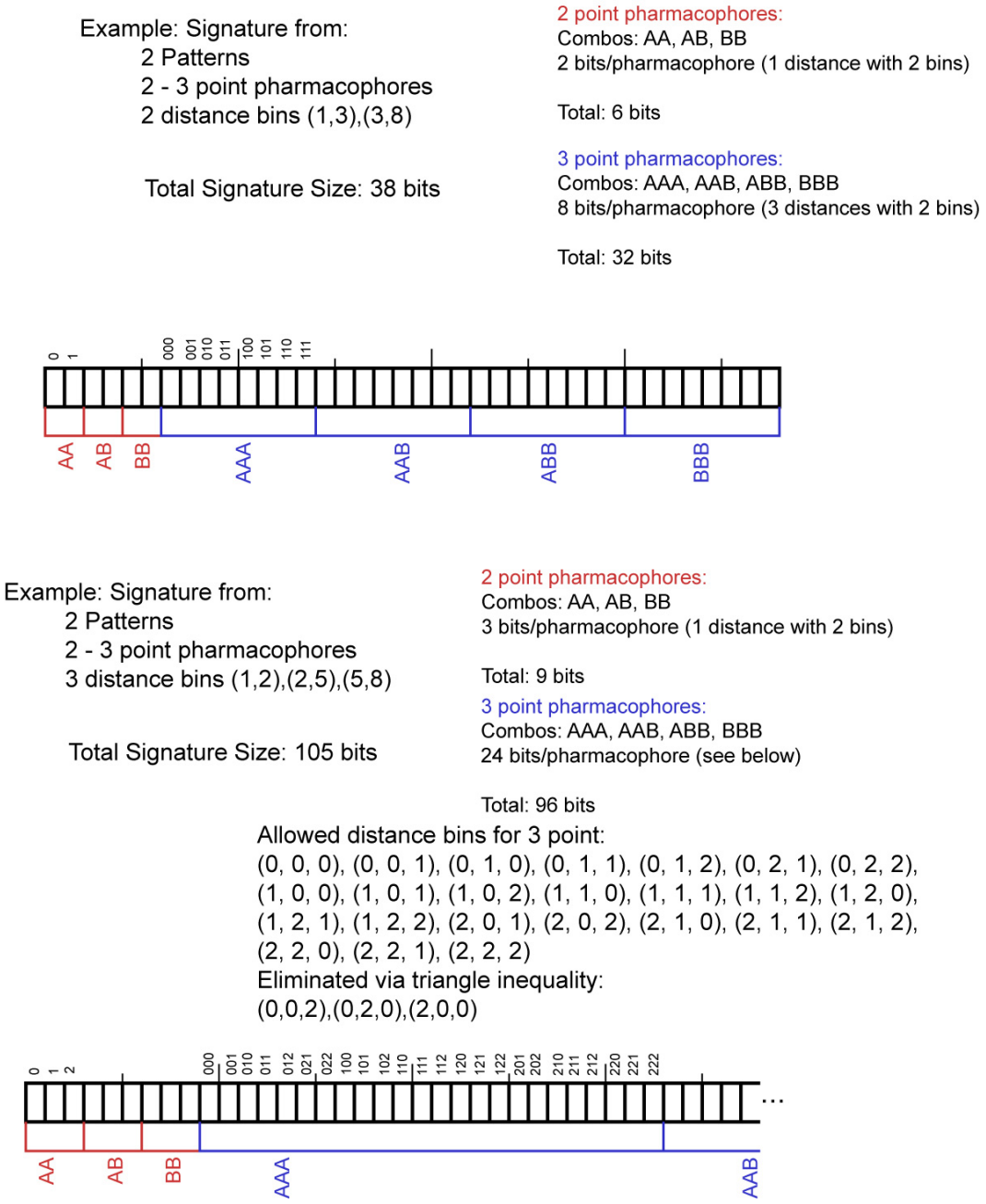
Farmakofor F je reprezentován dvojicí (P_F, D_F), kde P_F je reprezentace bodů a D_F reprezentace (zapřihrádkovaných) vzdáleností mezi body. Samotné P_F reprezentuje proto-farmakofor a samotné D_F reprezentuje lešení.

Proto-farmakofor složený z bodů $p_0, \ldots, p_{N-1}$ je reprezentován jako N-tice $(f_0, \ldots, f_{N-1})$, kde f_i je identifikátor vlastnosti i-tého bodu p_i, přičemž body $p_0, \ldots, p_{N-1}$ jsou uspořádány tak, aby $f_i \leq f_{i+1}$ pro $0 \leq i \leq N-1$, tedy vzestupně podle identifikátorů vlastností. Identifikátor vlastnosti je unikátní číslo z množiny $\{0, \ldots, f-1\}$, kde f je počet uvažovaných farmakoforových vlastností.

Pro reprezentaci vzdáleností ve farmakoforu není nutné si pamatovat všech $N \cdot (N-1)/2$ vzdáleností mezi jednotlivými body. Farmakofor definuje (topologický) N-úhelník (kde dvojúhelník je topologická úsečka), můžeme ho tedy jednoznačně reprezentovat triangulací tohoto N-úhelníku. To, jak je N-úhelník triangulován, je pevně dáno (jelikož obecně lze N-úhelník triangulovat vícero způsoby). Jak daná triangulace vypadá, které vzdálenosti jsou k reprezentaci farmakoforu použity a jak jsou uspořádány je popsáno na obrázku 3.1. Reprezentace vzdáleností D_F ve farmakoforu F je tedy $(2N-3)$-tice $(d_0, \ldots, d_{2N-4})$, kde d_i je vzdálenost mezi body p_0 a p_{i+1}, pokud je $i < N-1$. Pro $i \geq N-1$ je d_i vzdálenost mezi body p_{i-N+2} a p_{i-N+3}.



Obrázek 3.1: Triangulace farmakoforu jak je popsána v RDKitu. Body farmakoforu $p_0, \ldots, p_{N-1}$ jsou uspořádány vzestupně podle identifikátorů svých vlastností. K reprezentaci farmakoforu jsou použity vzdálenosti $d_0, \ldots, d_{2N-4}$, v tomto pořadí, přičemž d_i, pro $0 \leq i \leq N-2$, je vzdálenost mezi p_0 a p_{i+1} a pro $i \geq N-1$ je d_i vzdálenost mezi p_{i-N+2} a p_{i-N+3}. Povšimněme si, že vzdálenosti, které farmakofor reprezentují, jsou závislé na uspořádání bodů. Převzato z (Landrum, 2015b)



Obrázek 3.2: Příklad uspořádání farmakoforů v otisku. Otisky jsou vytvořené z farmakoforů, které se skládají ze dvou vlastností A a B, přičemž farmakofory mohou být dvou- nebo tříbodové. V horním příkladu jsou vzdálenosti přihrádkovány do dvou košů, v dolním do tří. Z otisku je možné vynechat farmakofory, jejichž vzdálenosti nesplňují trojúhelníkovou nerovnost – takové farmakofory nemohou existovat. Převzato z (Landrum, 2015a).

Máme tedy definovanou reprezentaci farmakoforu a můžeme přistoupit k samotnému otisku. Jak vypadá náš farmakoforový prostor? Uvažujme farmakofor o m až n bodech, f vlastnostech a b koších (pro přihrádkování vzdáleností). Potom každá N-tice (pro $m \leq N \leq n$) čísel mezi nulou a $f - 1$ uspořádaná vzestupně je validní proto-farmakofor. Stejně tak každá $(2N - 3)$-tice čísel mezi nulou a $b - 1$ je potenciální lešení. Některá z těchto lešení nemusí být „reálná“, neboť triangulace, kterou reprezentují, nesplňuje trojúhelníkovou nerovnost, ale RDKit umožňuje vytvářet otisky jak bez nich, tak s nimi. Každá dvojice proto-farmakoforu a lešení pak reprezentuje nějaký farmakofor.

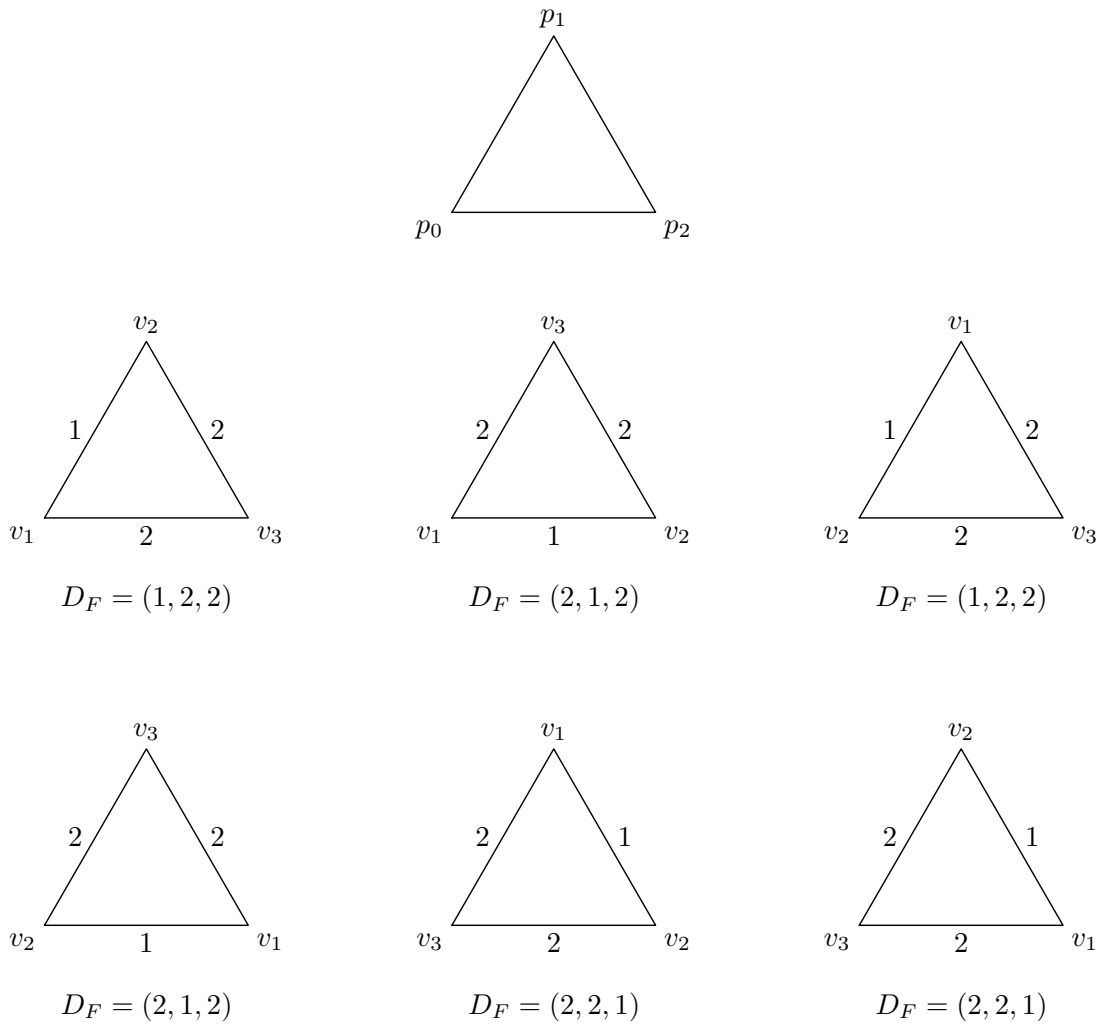
Každý farmakofor z našeho farmakoforového prostoru je reprezentován nějakým bitem. Čísla bitů jsou farmakoforům přidělena trojúrovňově. Nejdříve jsou farmakofory rozděleny podle počtu bodů – ty s nižším počtem bodů mají bity s nižšími čísly (tj. dvoubodové budou mít nižší bity než ty tříbodové). Farmakofory se stejným počtem bodů jsou dále rozděleny podle vlastností jejich bodů, přičemž pro farmakofory F a G platí, že pokud P_F je lexikograficky menší[1] než P_G, pak F dostane nižší bit než G. Tedy farmakofory s body (0,1,1) budou mít nižší bity než ty s body (0,1,2). Nakonec, farmakoforům se stejnými reprezentacemi bodů jsou přiděleny bity podle vzdáleností. Farmakofor F dostane nižší bit než farmakofor G, pokud D_F je lexikograficky menší než D_G. Farmakofor se vzdálenostmi (0,1,0) tedy dostane nižší bit, než farmakofor se vzdálenostmi (1,0,0). Na obrázku 3.2 je příklad toho, které bity budou reprezentovat které farmakofory.

Celá reprezentace farmakoforů má ale háček – není jednoznačná. Uvažme tříbodový farmakofor F skládající se ze tří bodů v_1, v_2, v_3, které mají stejnou vlastnost (přidělíme jí číslo nula), přičemž vzdálenost mezi v_1 a v_2 je jedna a vzdálenosti mezi v_1 a v_3 i mezi v_2 a v_3 jsou dva. Jak bude takový farmakofor reprezentovaný? Všechny body mají číslo nula, tedy $P_F = (0,0,0)$. Jak ale budou vypadat vzdálenosti? Pokud zvolíme $p_0 = v_1$, $p_1 = v_2$ a $p_2 = v_3$, pak $D_F = (1,2,2)$. Pokud ale zvolíme $p_0 = v_3$, $p_1 = v_2$ a $p_2 = v_1$ (což můžeme, neboť vlastnosti budou stále uspořádány vzestupně podle svých čísel), pak $D_F = (2,2,1)$! Jedna molekula tak může mít několik různých otisků, což jistě není dobré.

Proto je definováno *kanonické uspořádání* bodů, které definuje, jak mají být body seřazeny, pokud uspořádání pomocí identifikátorů vlastností není jednoznačné. V dokumentaci RDKitu o tomto uspořádání není žádná zmínka, při prozkoumání zdrojových kódů lze však narazit na část kódu, která kanonické uspořádání realizuje, z ní je možné vyčíst jeho definici. Kanonické uspořádání bodů ve farmakoforu F je takové, že odpovídající P_F je validní reprezentace proto-farmakoforu (tj. identifikátory vlastností bodů jsou uspořádány vzestupně) a zároveň odpovídající D_F je lexikograficky největší možné. Pro farmakofor z předchozího odstavce je kanonické uspořádání v_3, v_2, v_1, neboť odpovídající D_F je (2,2,1), což je lexikograficky větší, než (1,2,2) i (2,1,2) (viz obrázek 3.3). Ani kanonické uspořádání nedává jednoznačné pořadí bodů ve farmakoforu (stejně tak uspořádání v_3, v_1, v_2 dá $D_F = (2,2,1)$), ale už dává jednoznačnou reprezentaci farmakoforu.

Právě kanonické uspořádání je to jediné, co brání podpoře více než tříbodových farmakoforů. RDKit totiž implementuje kanonické uspořádání pouze pro ně (pro dvoubodové není třeba).

[1] N-tice $(a_1, \ldots, a_N)$ je lexikograficky menší než N-tice $(b_1, \ldots, b_N)$, pokud existuje pozice i taková, že $a_j = b_j$ pro $j < i$ a zároveň $a_i < b_i$.



Obrázek 3.3: Farmakofor F se skládá ze tří bodů v_1, v_2, v_3, které mají stejnou vlastnost s identifikátorem nula. Na obrázku jsou všechna možná uspořádání bodů (jelikož všechny body mají stejnou vlastnost, jsou možná všechna uspořádání) a jim odpovídající reprezentace vzdáleností D_F. Pouze poslední dvě uspořádání v dolním řádku jsou kanonická.

3.3 Původní implementace generování otisků

Než se začneme zabývat tím, jak umožnit podporu obecných N-bodových farmakoforů, musíme popsat, jak je celé generování implementováno. Nebudeme ovšem zabíhat do podrobností, pokud to nebude nutné pro implementaci našeho rozšíření. Speciálně některé techničtější části implementace jen zmíníme.

Celá tvorba otisku začíná zavoláním funkce `Gen2DFingerprint`, která jako argument dostane molekulu `M`, jejíž otisk má vytvořit, a instanci třídy `SigFactory`, která obsahuje všechny parametry. Nejdříve je vytvořena matice nejkratších vzdáleností mezi všemi atomy v `M`. Následně je na základě parametrů ze `SigFactory` vytvořen seznam všech možných reprezentací proto-farmakoforů – tedy seznam všech možných m až n-tic (m je minimální počet bodů ve farmakoforu a n maximální) z čísel $0, \ldots, f-1$, kde f je počet farmakoforových vlastností.

Poté jsou nalezeny výskyty všech vlastností v molekule `M`. *Výskytem vlastnosti* myslíme konkrétní atom, či skupinu atomů, které mají danou vlastnost. Výskytem vlastnosti „aromatické jádro“ mohou být například atomy tvořící benzenové jádro, výskytem vlastnosti „donor vodíkového můstku“ může být například atom kyslíku v OH skupině. Na základě výskytů vlastností jsou pro každou reprezentaci proto-farmakoforu vytvořeny všechny její instance v molekule `M`. Tedy pro proto-farmakofor s reprezentací (0,0,1) nalezneme všechny trojice (a_1, a_2, a_3) atomů (či skupin atomů) z `M` takové, že a_1 a a_2 jsou výskyty vlastnosti s identifikátorem nula a a_3 je výskyt vlastnosti s identifikátorem jedna. Z každé takové instance proto-farmakoforu můžeme vytvořit odpovídající farmakofor.

Každá instance proto-farmakoforu je spolu s maticí nejkratších vzdáleností a bitovým vektorem otisku předána funkci `_ShortestPathMatch`. Jejím úkolem je vytvořit odpovídající farmakofor (spočítáním vzdáleností mezi body – skupinami atomů) a nastavit v otisku bit reprezentující tento farmakofor na jedničku.

Funkce `_ShortestPathMatch` nejprve provede triangulaci a spočítá příslušné vzdálenosti mezi body v instanci proto-farmakoforu. Tím vznikne farmakofor, který předá funkci `SigFactory.GetBitIdx`. Ta nejdříve reprezentaci farmakoforu převede do kanonického tvaru a poté spočítá, který bit v otisku přísluší farmakoforu. Funkce `_ShortestPathMatch` následně tento bit nastaví na jedničku.

3.4 Rozšíření implementace

Abychom umožnili generovat otisky s více než tříbodovými farmakofory, bylo třeba naimplementovat dvě věci. Jednak již zmíněné kanonické uspořádání farmakoforů s více než třemi body, a pak také generování vzdáleností v triangulaci pro libovolně velké farmakofory. Nejdříve začneme právě s triangulací, neboť se jedná o velmi malou změnu.

Pro zjištění, mezi jakými body v N-bodovém farmakoforu je třeba spočítat vzdálenost, slouží datová struktura (konkrétně slovník), ve které je pro N mezi dvěma a deseti seznam hledaných dvojic bodů uložen. Nad touto datovou strukturou jsme proto vytvořili obal (neboť jde o vestavěnou datovou strukturu Pythonu), který vygeneruje seznam vzdáleností, když je vyžádán takový, který dosud nebyl uložen. Samotné vygenerování dvojic bodů vychází přímo ze vzorce uvedeného v kap. 3.2.

Nyní ke kanonickému uspořádání. Označme si množinu bodů ve farmakoforu jako $B = \{b_0, \ldots, b_{N-1}\}$, f_i pak bude identifikátor vlastnosti bodu b_i, přičemž body jsou očíslovány tak, že pro $i < j$ platí $f_i \leq f_j$. Dále p_i bude značit i-tý bod v kanonickém uspořádání. Nakonec si ještě označíme $D = \{d_0, \ldots, d_{2N-4}\}$ množinu vzdáleností, které reprezentují farmakofor, d_i pro i mezi nulou a $N-2$ je vzdálenost mezi p_0 a p_{i+1} pro $i \geq N-1$ je to vzdálenost mezi p_{i-N+2} a p_{i-N+3}.

Kanonické uspořádání bodů ve farmakoforu budeme hledat rekurzivně. V i-té úrovni rekurze budeme maximalizovat vzdálenost d_i. V nulté úrovni to je vzdálenost mezi body p_0 a p_1, v první úrovni mezi p_0 a p_2 a tak dále.

Vzdálenost maximalizujeme vhodnou volbou bodů, mezi nimiž ji počítáme. V nulté úrovni chceme tedy vhodně zvolit body p_0 a p_1, v první bod p_1 (neboť bod p_0 už je určený) atd. Všimněme se, že rekurzi můžeme skončit nejpozději na $(N-1)$-ní úrovni, neboť prvních $N-1$ vzdáleností obsahuje všechny body – každý bod z B tedy bude mít určenou pozici v kanonickém uspořádání.

Body ovšem nemůžeme volit úplně libovolně. Můžeme použít jen ty, které ještě nemají přidělené pořadí v kanonickém uspořádání – množinu takových bodů budeme značit B'. Potom musíme udržet body v kanonickém uspořádání seřazené vzestupně podle identifikátorů jejich vlastností. Kandidátem pro neurčený bod p_i je množina bodů K_i, která obsahuje body z B' takové, že mají vlastnost s identifikátorem f_i. Je-li bod p_i už určený jako b_j, pak množina kandidátů K_i bude obsahovat pouze b_j.

Máme-li kandidáty, můžeme najít všechny validní dvojice bodů, které maximalizují danou vzdálenost. Předpokládejme, že maximalizujeme vzdálenost mezi p_i a p_j. Najdeme všechny dvojice bodů k_i a k_j takové, že $k_i \in K_i$, $k_j \in K_j$ a vzdálenost mezi k_i a k_j je maximální možná. Pro každou takovou dvojici položíme $p_i = k_i$, $p_j = k_j$, oba body odstraníme z B' a rekurzivně určíme zbytek uspořádání. Ze všech takto získaných uspořádání pak vybereme to, které má posloupnost vzdáleností $d_0, \ldots, d_{2N-4}$ lexikograficky největší.

Celý algoritmus v pseudokódu vypadá následovně:

```
1  N -> počet bodů ve farmakoforu
2  D -> D[j] je dvojice (p1,p2) pozic bodů v kan. usp. vzdálených di
3  F -> F[p] pro bod p je identifikátor jeho vlastnosti
4
5  def all_distances(
6      P -> seznam bodů
7      D -> seznam dvojic bodů
8  ){
9      return [distance((P[p1],P[p2])) for (p1,p2) in D]
10 }
11
12 def BestOrder(
13     i -> číslo vzdálenosti, kterou maximalizujeme
14     B' -> množina bodů bez přiděleného pořadí
15     P -> seznam bodů, jež mají přidělené pořadí,
16          P[j] je j-tý bod v kan. usp.
17 ){
18
19
```

```
20      // konec rekurze - všechny body mají kan. pořadí
21      if (i == N - 1)
22          return P
23
24      // body, mezi nimiž max. vzdálenost
25      p1 = D[i][0]
26      p2 = D[i][1]
27
28      // najdeme kandidáty pro p1 a p2
29      if (p1 in P)
30          K1 = [p1]
31      else
32          K1 = [p for p in B' if (F[p] == F[p1])]
33      // p2 nikdy nemá určené pořadí
34      K2 = [p for p in B' if (F[p] == F[p2])]
35
36      // najdeme dvojice maximalizující vzdálenost mezi p1 a p2
37      opt = []
38      maxDist = -1
39      for (k1 in K1, k2 in K2){
40          d = distance(k1,k2)
41          if (d > maxDist){
42              opt = [(k1,k2)]
43              maxDist = d
44          }
45          else if(d == maxDist)
46              opt.add((k1,k2))
47      }
48      // pro každou dvojici z opt rekurzivně dopočítáme zbytek pořadí
49      orderings = []
50      for ((k1,k2) in opt){
51          P[p1] = k1
52          P[p2] = k2
53          B'' = B'.remove(k1,k2)
54          o = BestOrder(i+1,P,B'')
55          orderings.add(o)
56      }
57      // vybereme uspořádání z orderings s lexikograficky největšími vzdálenostmi
58      best = orderings[0]
59      maxDists = all_distances(best, D)
60      for (o in orderings){
61          dists = all_distances(best, D)
62          if (dists lex_greater maxDist){
63              best = o
64              maxDists = dists
65          }
66      }
67      return best
68  }
69
70  KanonickéUspořádání = BestOrder(0,B,[])
```

Takto získané uspořádání bude určitě kanonické. Funkce `BestOrder` má vrátit uspořádání bodů, které má lexikograficky největší vzdálenost, přičemž jeho začátek je určený parametrem P. Je-li i rovno $N-1$, pak je určeno celé uspořádání, takže funkce opravdu vrátí, co má. Pro menší i máme určeno prvních i vzdáleností, i-tou vzdálenost zvolíme jako maximální možnou, dopočítáme možné hodnoty zbylých vzdáleností a vybereme maximální (lexikograficky) – získali jsme tedy maximální možné vzdálenosti. Odpovídající pořadí bodů je tudíž kanonické.

3.5 Optimalizace

Během experimentů se ukázalo, že generování otisků je velmi náročné na výpočetní zdroje, obzvláště na výpočetní čas. Vytvoření dvou- až tříbodových otisků pro dataset o 15 000 molekulách trvalo zhruba osm hodin[2]. Vzhledem k tomu, že počet farmakoforů (a s tím nutně i čas potřebný ke spočítání otisku) roste exponenciálně s počtem bodů[3], rozhodli jsme se prozkoumat možné optimalizace generování.

Funkci `Gen2DFingerprint` jsme analyzovali pomocí nástroje `cProfile`, který je k dispozici pro Python. Ze získaných údajů vyplynulo, že zhruba 75 % času generování bylo stráveno ve funkce `GetUniqueCombinations` (nacházející se v modulu `Chem.Pharm2D.Utils`). Při jejím podrobném prozkoumání se ukázalo, že je napsaná velmi neefektivně, proto jsme se rozhodli ji upravit. Tím se nám povedlo ji zrychlit – při použití naší implementace je ve ní stráveno jen zhruba 16 % z celkového času (přičemž ostatní funkce zůstaly nezměněny), viz obrázek 3.4. Toto zrychlení nám umožnilo provést experimenty i s pětibodovými otisky, což by jinak nebylo reálné.

Cílem funkce `GetUniqueCombinations` je pro danou n-tici, ne nutně různých vlastností (tj. reprezentaci proto-farmakoforu), nalézt všechny n-tice, které obsahují výskyty těchto vlastností v molekule. Jako první argument dostává funkce seznam n (ne nutně různých) *tříd* C. Třída (prvek C) je seznam atomů (či skupin atomů), které mají danou vlastnost. Prvky třídy mají navíc jednoznačně určené pořadí. Druhý argument je seznam I, který obsahuje identifikátory tříd, tedy pokud $I[i] = I[j]$, pak $C[i] = C[j]$. Navíc stejné třídy v C jsou „pohromadě“, tedy identifikátor seznamu tříd může být například (0,0,1,2) nebo (0,1,2,2), ale ne (0,2,1,2).

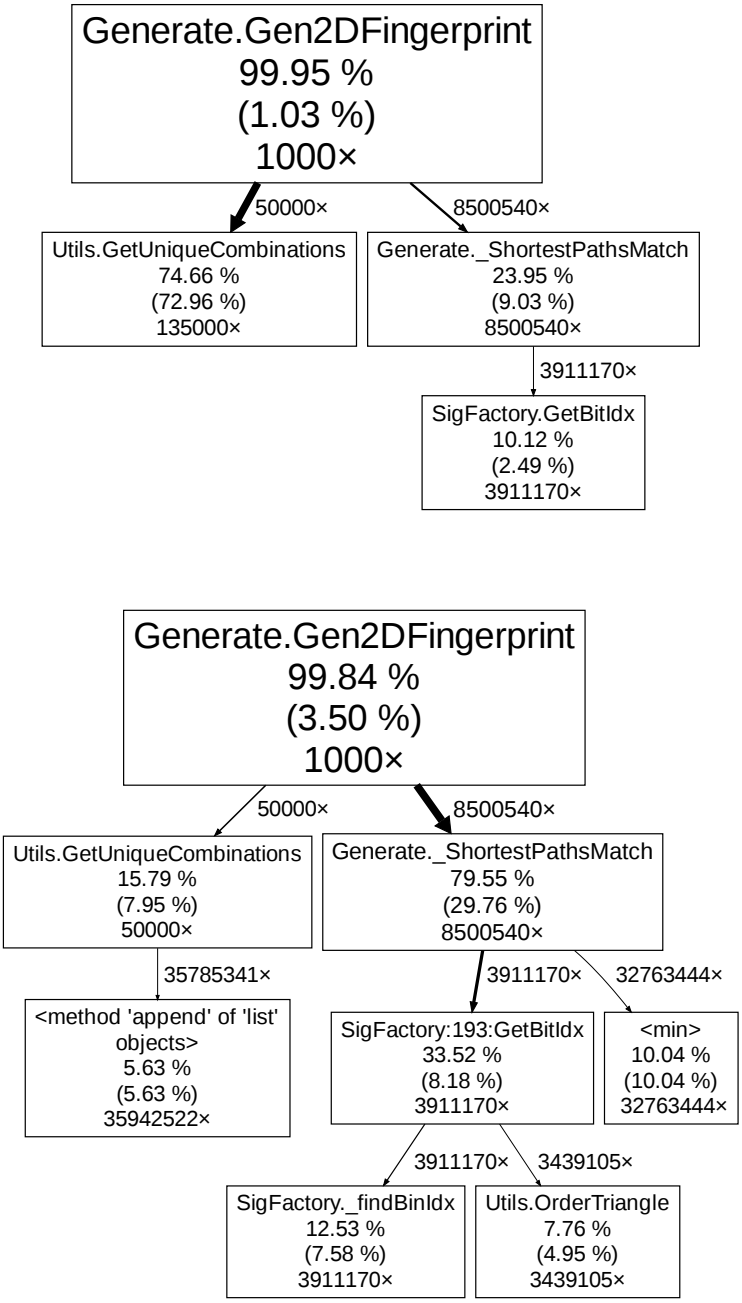
Výstupem je seznam $\mathcal{S}$ obsahující všechny *výběry*. Výběr je n-tice S taková, že $S[i] \in C[i]$. Navíc pro každý výběr S musí platit:

1. Žádné dva prvky S nejsou stejné. Tedy pro každé i, j platí, že pokud $S[i] = S[j]$, tak $i = j$.

2. Prvky S pocházející ze stejné třídy jsou uspořádány stejně jako ve třídě. Formálně, nechť $S[i] = T[a]$ a $S[j] = T[b]$, kde $T \in C$. Potom $a \geq b \Rightarrow i \geq j$.

[2]Použit byl notebook Asus N61Vg s procesorem Intel Core2 Duo T6600 taktovaným na 2,2 GHz, 4 GB RAM a operačním systémem Debian Wheezy.

[3]Pro srovnání: čistě dvoubodových farmakoforů je s naší volbou vlastností 45, dvou- až tříbodových je 990, dvou- až čtyřbodových je 18 000 a dvou- až pětibodových je více než 293 000.



Obrázek 3.4: Analýza efektivity funkce `Gen2DFingerprint` pomocí nástroje `cProfile`. Horní diagram je pro původní implementaci funkce `GetUniqueCombinations`, dolní pro naši implementaci. Obdélníky odpovídají jednotlivým funkcím a šipky volání funkcí. První číslo v obdélníku u funkce je celkový čas strávený v ní (v procentech z celkové doby běhu programu), v závorce potom bez započítání funkcí z ní volaných. Poslední číslo udává, kolikrát byla funkce zavolána (včetně rekurzivních volání). Číslo u šipky z funkce f do funkce g udává počet volání funkce g funkcí f. Funkce, ve kterých bylo stráveno méně než 5 % celkového času nejsou zobrazeny. Měření bylo provedeno na prvním tisíci neaktivních molekul z MUV datasetu AID 466 (viz kapitola 5.1). Generovány byl dvouaž tříbodové otisky.

Příklad: Nechť $C[0] = (A, B)$, $C[1] = (A, B)$, $C[2] = (B, C)$, $I[0] = I[1] = 0$ a $I[2] = 1$. Pro tyto třídy existuje jen jediný validní výběr (A, B, C). Trojice (A, B, A) není validní, neboť obsahuje dva stejné prvky (první podmínka), (B, A, C) zase nesplňuje uspořádánání (druhá podmínka).

Původní implementace funkce `GetUniqueCombinations` byla značně neefektivní. Odebrala první třídu ze seznamu a rekurzivně se spustila na zbylé třídy. Poté se ke každému S, které bylo v seznamu vráceném z rekurze, pokusila přidat všechny možné prvky z odstraněné třídy $C[0]$. Každý takto vzniklý seznam bylo třeba správně setřídit (aby byly prvky ze stejné třídy ve správném pořadí) a bylo nutné zkontrolovat, zda nebyly vytvořeny dva stejné seznamy.

Naše vylepšení vychází z toho, že můžeme výběry rozdělit na části, přičemž každá bude obsahovat prvky z jedné třídy (tj. výběr z tříd s identifikátory (0,1,1,2) rozdělíme na tři části (0), (1,1), (2)). Vytvořit všechny výběry k prvků, splňující podmínky 1. a 2., z jedné třídy je pak snadné.

Budeme mít pomocnou funkci `CombosOfClass`, která dostane jako argument třídu T a číslo k a vytvoří seznam všech k-tic prvků z T takových, že neobsahují dva stejné prvky a jsou seřazené stejně jako třída T. Funkce bude rekurzivní. Nechť x je poslední prvek T. Funkce se rekurzivně zavolá na T bez x a k, čímž vytvoří seznam O všech k-tic neobsahujících x, poté se zavolá na T bez x a $k-1$, čímž vznikne seznam W všech $(k-1)$-tic bez x. Následně na konec každé $(k-1)$-tice z W přidá x, čímž vzniknou všechny k-tice obsahující x. Spojení seznamů O a W je kýžený výsledek. Rekurze se zastaví, když $k = 1$ (každý prvek T tvoří výběr), nebo když k je rovno počtu prvků v T (celé T je výběr).

V pseudokódu funkce `CombosOfClass` vypadá následovně:

```
1  def CombosOfClass(
2      T -> třída, seznam nějakých prvků
3      k -> kolik prvků z T tvoří výběr
4  ){
5      s = T.size
6
7      // konec rekurze
8      if (k > s or s <= 0 or k <= 0)   // žádný takový výběr není
9          return []
10     else if (k == s)   // jediný takový výběr je celé T
11         return T
12     else if (k == 1)   // každý prvek T tvoří výběr
13         return [[x] for x in T]
14
15     // poslední prvek
16     x = T.last_elem()
17     withoutLast = CombosOfClass(T.remove(x),k)
18     withLast = CombosOfClass(T.remove(x),k-1))
19     for (c in withLast)
20         c.add(x)
21
22     return withLast + withoutLast
23 }
```

Samotná funkce `GetUniqueCombinations` nejdříve pro každou třídu T_i z C zjistí počet jejích výskytů k_i v C. Pak pro každou T_i zavolá `CombosOfClass` s argumenty T_i a k_i, výsledný seznam výběrů si označíme S_i. Tyto seznamy spojíme, tj. každý prvek z S_0 zřetězíme s každým prvkem S_i a tak dále. Výsledek si označíme $\mathcal{S}$. Všechny prvky $\mathcal{S}$ jsou správně uspořádané výběry ze tříd C, jen se v nich mohou vyskytovat stejné prvky (pokud dvě různé třídy obsahují společný prvek). Musíme tedy ještě $\mathcal{S}$ projít a všechny takové výběry z něj smazat.

Pseudokód pro funkci `GetUniqueCombinations` bude vypadat následovně:

```
1  def GetUniqueCombinations(
2      C -> seznam tříd
3      I -> identifikátory tříd
4  ){
5      n = C.size
6
7      // výsledný seznam
8      res = [[]]
9
10     i = 0
11     while (i < n){
12         // třída, pro níž chceme vytvořit všechny výběry
13         T = C[i]
14         classId = I[i]
15
16         // spočítáme všechny výskyty T v C (jsou za sebou)
17         k = 0
18         while (I[i] = classId)
19             k++, i++
20
21         // najdeme všechny k-tice prvků z T
22         S = CombosOfClass(T,k)
23
24         // spojíme prvky S a res, každý s každým
25         tmp = []
26         for (x in S, y in res)
27             tmp.add(x + y)
28         res = tmp
29     }
30
31     // zkontrolujeme, zda se nějaký prvek nevyskytuje vícekrát
32     for (sel in res){
33         for (i = 0; i < sel.size; i++){
34             if (sel[i] in sel[:i]){
35                 res.remove(sel)
36                 break
37             }
38         }
39     }
40     return res
41 }
```

4. Korelační analýza

Jedním z potenciálních problémů reprezentace molekul pomocí farmakoforových otisků jsou závislosti mezi farmakofory. Některé farmakofory jsou si totiž velmi podobné (pokud se například liší jen v jednom bodu), nebo dokonce jeden farmakofor může být „podmnožinou“ jiného. Například dvoubodový farmakofor s vlastnostmi A a B vzdálenými jednu jednotku je součástí tříbodového farmakoforu s vlastnostmi A, A, B se vzdáleností A-A rovnou dvěma a vzdálenostmi A-B jedna a dvě jednotky. Může se tedy stát, že na některých datech budou výskyty některých farmakoforů v molekulách silně provázané – výskyt farmakoforu F bude znamenat i výskyt farmakoforu G a naopak.

Takové závislosti pak mohou způsobovat nežádoucí zvýšení významu některých farmakoforů. Pokud se dva farmakofory vyskytují zásadně společně, pak je velká pravděpodobnost, že se jedná jen o jeden farmakofor, na který se díváme dvěma způsoby. Tím jsme mu ale přiřadili větší význam než ostatním farmakoforům, neboť se bude v naší reprezentaci vyskytovat dvakrát.

Ne všechny typy závislostí ale způsobují redundanci ve farmakoforovém modelu. Například, pokud se F vyskytuje všude tam kde G, ale ne naopak, pak G může být signifikantní, aniž by byl signifikantní F. Budeme se proto zabývat jen závislostmi, kde se dva či více farmakoforů vyskytuje společně – je-li jeden z nich signifikantní, pak jsou signifikantní všechny.

V této kapitole popíšeme rozšíření 2D farmakoforové metody, jehož cílem je identifikovat takovéto závislosti mezi farmakofory a následně se jich zbavit. To by teoreticky mělo vést k lepšímu výkonu celé metody.

4.1 Určení závislosti

Dva farmakofory F a G bychom chtěli prohlásit za *závislé*, pokud se v molekulách vyskytují společně, tedy buď oba, nebo žádný z nich. Tato definice závislosti je velmi striktní pro použití na reálných datech, neboť pravděpodobnost, že se dva různé farmakofory budou vždy vyskytovat společně, je malá. Je tedy třeba míru závislosti dvou farmakoforů nějak kvantifikovat a poté nastavit hranici, kdy prohlásíme dva farmakofory za závislé. Ještě podotkneme, že závislost farmakoforů tak, jak ji budeme používat, je závislá na datech (sadě molekul), se kterými pracujeme. Uvažovat závislosti obecně nemá velký smysl, protože není těžké vytvořit pro každou dvojici farmakoforů molekulu, která jeden obsahuje a druhý ne.

Pro určení míry závislosti farmakoforů v daných datech jsme se rozhodli použít *Pearsonův korelační koeficient pro populaci* (PPMCC – Pearson product-moment correlation coefficient). PPMCC pro dvě populace $x = \{x_1, \ldots, x_n\}$ a $y = \{y_1, \ldots, y_n\}$ je (podle Lee Rodgerse a Nicewandera (Lee Rodgers a Nicewander, 1988)) definován následovně:

$$r_{xy} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{x})^2}},$$

kde $\bar{x}$, resp. $\bar{y}$, je aritmetický průměr z x, resp. y. Je-li jeden ze součinů ve jmenovateli nulový, pak PPMCC není definován. Hodnota r_{xy} se vždy pohybuje v intervalu $\langle -1; 1 \rangle$. Absolutní hodnota $|r_{xy}|$ vyjadřuje míru lineární závislosti mezi x a y, znaménko potom její směr (tj. kladné znaménko znamená, že odpovídající lineární funkce je rostoucí, záporné, že je klesající).

Pro farmakofor F si jako $D(F)$ označíme vektor jedniček a nul, ve kterém $D(F)_i = 1$, pokud se F vyskytuje v i-té molekule, jinak $D(F)_i = 0$. Míra závislosti $Z(F,G)$ farmakoforů F a G buď potom rovna $r_{D(F)D(G)}$. Čím častěji se farmakofory F a G vyskytují v našich datech společně, tím vyšší bude hodnota $Z(F,G)$. Záporná hodnota $Z(F,G)$ naopak znamená, že výskyt jednoho z farmakoforů vylučuje výskyt druhého. Dva farmakofory tedy prohlásíme za závislé (či korelované, neboť výskyty těchto farmakoforů korelují), pokud jejich míra závislosti Z překročí určitou pevně danou hranici t. Číslu t budeme říkat *hranice závislosti*.

Problém však nastává, pokud PPMCC není pro danou dvojici farmakoforů F a G definován, tedy pokud je jeden ze součinů ve jmenovateli nulový (nechť je to ten pro F). Kdy může taková situace nastat? Ve dvou případech:

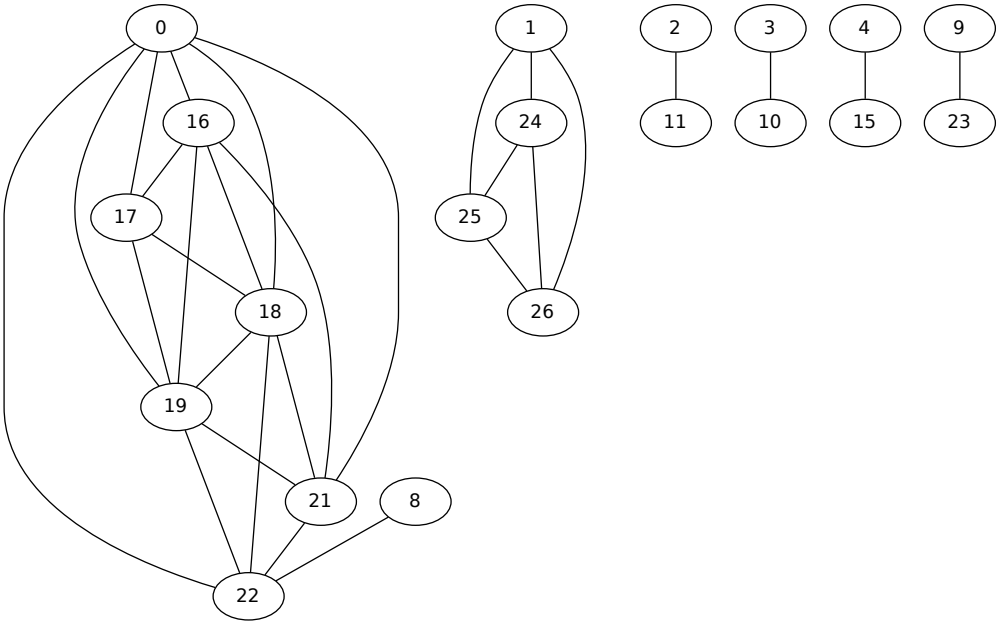
1. F se nevyskytuje v žádné molekule.
2. F se vyskytuje ve všech molekulách.

První případ je poměrně častý (kvůli kanonickému uspořádání nebo trojúhelníkové nerovnosti nejsou některé bity v otisku nikdy nastaveny na jedničku). Vzhledem k tomu, že se F nikdy nedostane do farmakoforového modelu (neboť není pravda, že by se vyskytoval signifikantně více v aktivních molekulách), a tedy neovlivní skóre, jsme se rozhodli v takovém případě neprohlásit F a G za závislé. Druhý případ je naopak velmi vzácný, ale podobný, neboť F taktéž nemůže být signifikantní a tedy také neovlivní skóre. Proto jsme se i ve druhém případě rozhodli prohlásit F a G za nezávislé.

4.2 Odstranění závislostí

Nyní, když máme definováno, kdy jsou farmakofory závislé, můžeme přejít k tomu, jak závislosti odstranit. Nejdříve se nám bude hodit nějaká reprezentace závislostí. Tou bude *korelační graf* – neorientovaný graf, jehož vrcholy budou jednotlivé farmakofory a hrana povede mezi těmi vrcholy, jimž odpovídají závislé farmakofory (závislost farmakoforu se sebou samým ignorujeme).

V ideálním případě by závislosti měly být tranzitivní, tedy pokud A a B jsou závislé (tj. vyskytují se zásadně společně) a B a C jsou závislé, pak by i A a C měly být závislé. V takovém případě by se korelační graf skládal z několika nezávislých úplných grafů (grafů, kde je hranou spojena každá dvojice vrcholů). Každý takový úplný graf by odpovídal jednomu „superfarmakoforu“, na který se díváme vícero způsoby. Pro odstranění závislostí by pak stačilo z každého „superfarmakoforu“ vzít jednoho reprezentanta a zbylé smazat.



Obrázek 4.1: Korelační graf signifikantních farmakoforů. Zatímco podgrafy napravo odpovídají naší představě o tranzitivitě závislostí, podgraf nalevo úplný není.

Ve skutečnosti ovšem závislosti tranzitivní nejsou. Přesto je korelační graf obvykle složený z několika skupin vrcholů, přičemž vrcholy jsou v rámci skupin poměrně hustě propojené (viz obrázek 4.1). Algoritmus pro odstranění závislostí jsme tedy zvolili následovně: Najdeme v korelačním grafu vrchol v, který má nejvyšší stupeň[1] a ten prohlásíme za reprezentanta. Následně z korelačního grafu smažeme v a všechny jeho sousedy a celý postup opakujeme, dokud není korelační graf prázdný. Reprezentanti potom odpovídají farmakoforům, které budeme při screeningu používat, ostatní farmakofory (jejichž vrcholy jsme smazali z grafu) budeme ignorovat.

Tento algoritmus nám zajistí, že opravdu odstraníme všechny závislosti, tedy žádní dva reprezentanti nejsou závislí. To je proto, že při vybrání reprezentanta vždy odstraníme všechny s ním sousedící vrcholy. Dále pak bude platit, že každý farmakofor je buď přímo reprezentant (a tedy se účastní screeningu přímo), nebo existuje závislost mezi ním a nějakým reprezentantem (a tedy se screeningu zúčastní nepřímo).

Ještě je třeba zmínit jeden problém. Pokud bychom prováděli odstranění závislostí na všech farmakoforech, mohlo by dojít k tomu, že nějaký signifikantní farmakofor bude vyřazen, ale jeho reprezentant nebude signifikantní. Tím bychom přišli o farmakofor, který by měl být ve farmakoforovém modelu. Tento problém jsem se rozhodli vyřešit tím, že odstraňujeme závislosti jen mezi signifikantními farmakofory. Toto řešení jsme vybrali hlavně kvůli efektivitě – odstranit závislosti mezi všemi farmakofory vyžaduje spočítat korelační koeficient pro každou dvojici farmakoforů. Pro dvou- až pětibodové farmakofory, kterých je více než 290 000,

[1]Stupeň vrcholu v je počet vrcholů, do kterých vede z v hrana.

může odstranění závislostí trvat i několik desítek dní. Naopak signifikantních farmakoforů je jen zlomek z celkového počtu farmakoforů a tedy celý výpočet trvá řádově kratší dobu.

Následuje pseudokód celého odstraňování závislostí.

```
M -> seznam otisků všech molekul
F -> seznam všech farmakoforů
t -> minimální hodnota kor. koef., aby byly dva farmakofory závislé

// nejdřív zjistíme ve kterých molekulách se nacházejí které farmakofory
for (f in F){
    for (i = 0; i < M.size; i++){
        if (f in M[i])
            D[f][i] = 1
        else
            D[f][i] = 0
    }
}

// vytvoříme korelační graf
for (f in F, g in F){
    if (PPMCC(D[f],D[g]) > t)
        neighbours[f].add(g)
}

// spočítáme každému vrcholu stupeň
for (f in F)
    degs[f] = neighbours[f].size

// samotné vybírání reprezentantů
representants = []
while (not degs.empty()){
    // najdeme vrchol s největším stupněm
    v = argmax(degs)
    d = degs[v]
    representants.add(v)
    delete degs[v]

    // smažeme sousedy
    for (n in neighbours[v]){
        for (m in neighbours[n])
            degs[m]--
        delete degs[n]
    }
}
```

5. Experimentální výsledky

V této kapitole ukážeme výsledky našich úprav 2D farmakoforové metody a srovnáme je jak s původní metodou, tak i s některými jinými metodami. Nejprve představíme data, na kterých jsme prováděli experimenty a způsob, jakým jsme měřili „výkon“ metody. Poté jednotlivé experimenty, které jsme provedli, a jejich parametry. Nakonec ukážeme samotné výsledky experimentů a jejich interpretaci.

Celkem jsme provedli tři sady experimentů. Cílem první bylo zjistit, zda má odstranění závislostí vliv na výkon metody, a případně prozkoumat, jak moc je výkon ovlivněn hranicí, která určuje, zda jsou farmakofory závislé. Druhá sada experimentů měla za cíl prozkoumat, zda výkon metody poroste s rostoucím počtem bodů v použitých farmakoforech. Nakonec, ve třetí sadě jsme zkoumali, zda je možné pomocí farmakoforového modelu zlepšit výkon čistě otiskového screeningu.

5.1 Data

Pro experimenty jsme zvolily celkem dvě sady dat. První z nich jsou MUV (Maximum Unbiased Validation) datasety (Rohrer a Baumann, 2009), které použili Hoksza a Škoda ve svých experimentech. Cílem MUV metody bylo vytvořit datasety pro objektivní testování LBVS. Všechny MUV datasety pochází ze sloučenin ve veřejně dostupné databázi PubChem. Každý dataset obsahuje 30 aktivních a 15 000 neaktivních (decoys) molekul, které byly vytvořeny tak, aby byly rozprostřeny po chemickém prostoru tvořeném jednoduchými deskriptory. Jednoduché deskriptory Rohrer a Baumann (2009) definují jako „vektory obsahující celkový počet atomů, počet těžkých atomů, atomů bóru, bromu, [...] v molekule, počet akceptorů a donorů vodíkového můstku, logP, počet chirálních center a počet cyklických systémů“.

Druhou sadou dat byla kompilace datasetů, kterou použili Hoksza a Škoda v jiném článku (Hoksza a Škoda, In-press). Jedná se o datasety sebrané z různých zdrojů, které byly sestaveny podle „obtížnosti“ – průměrné hodnoty AUC (viz kap 5.2.1), které na ní dosahovaly běžně používané metody – AP, TT, MACCS keys, ECFP4 a FCFP4 (varianta ECFP4).

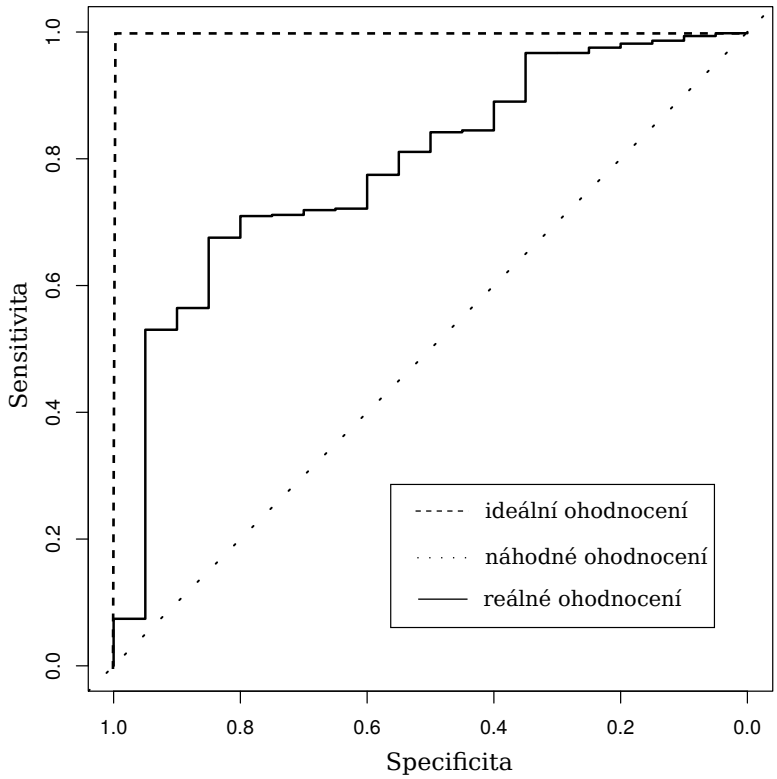
Z každého datasetu jsme náhodně vybrali část molekul, která byla určená jako známé molekuly (trénovací sada), a část, na které jsme provedli samotný screening (testovací sada). V případě MUV datasetů se trénovací sada skládala z deseti aktivních molekul a 3 000 neaktivních molekul, zbylé molekuly (dvacet aktivních a 12 000 neaktivních) byly použity jako testovací sada. U druhé sady bylo do trénovací sady vybráno třicet aktivních a sto neaktivních molekul, do testovací potom dvacet aktivních a 4 900 neaktivních.

5.2 Vyhodnocení

Všechny naše experimenty probíhaly podle následujícího vzoru:

1. Na základě trénovací sady molekul je vytvořen farmakoforový model.
2. Samotný screening molekul z testovací sady – pro každou je určeno její skóre (S_{pp} definované v kap. 2.3, není-li řečeno jinak).
3. Ze skóre molekul je vytvořena ROC křivka a vypočítáno AUC.

Pro každý dataset byl tento postup opakován desetkrát, pokaždé s jinou trénovací a testovací sadou molekul. Výsledná AUC byla nakonec zprůměrována. Celý systém, kterým jsme naše modifikace vyhodnocovali, je v Příloze 3, veškerá naměřená data jsou potom v Příloze 1.



Obrázek 5.1: Příklad ROC křivky. Čárkovaně je znázorněna „ideální“ křivka, kdy všechny aktivní molekuly mají vyšší skóre než ty neaktivní. Naopak tečkovaně je křivka odpovídající náhodnému ohodnocení databáze.

5.2.1 ROC křivky

K samotnému kvantifikování výkonu celé metody jsme použili *ROC (receiver operating characteristic) křivky*. ROC křivka (obrázek 5.1) dává do vztahu senzitivitu a specificitu metody (Triballeau a kol., 2005). Máme-li databázi molekul ohodnocenou virtuálním screeningem, můžeme určit nějakou hranici a molekuly

se skóre nad touto hranicí budeme považovat za potenciálně aktivní. Senzitivita vyjadřuje, kolik skutečně aktivních molekul jsme označili jako aktivní (*true positives* – TP) a kolik jsme jich označili jako neaktivní (*false negatives* – FN). Přesně je senzitivita vyjádřena jako

$$Se = \frac{N_{\mathrm{TP}}}{N_{\mathrm{TP}} + N_{\mathrm{FN}}},$$

kde N_{TP}, resp. N_{FN} je počet vybraných aktivních, resp. nevybraných aktivních, molekul. Naopak specificita vyjadřuje množství skutečně neaktivních molekul, které jsme vyřadili (*true negatives* – TN), tedy

$$Sp = \frac{N_{\mathrm{TN}}}{N_{\mathrm{TN}} + N_{\mathrm{FP}}},$$

kde N_{TN}, resp. N_{FP} je počet vyřazených, resp. nevyřazených, neaktivních molekul.

ROC křivka není nic jiného než graf závislosti Se na $1 - Sp$, tedy vztah mezi množstvím šumu $(1 - Sp)$ a množstvím užitečné informace (Se) v něm. Výkon VS je potom možné měřit jako obsah plochy pod ROC křivkou (AUC – Area Under Curve). Čím je AUC vyšší, tím víc se ROC křivka blíží ke křivce ideální metody, která dá všem aktivním molekulám vyšší skóre než těm neaktivním.

5.2.2 Parametry

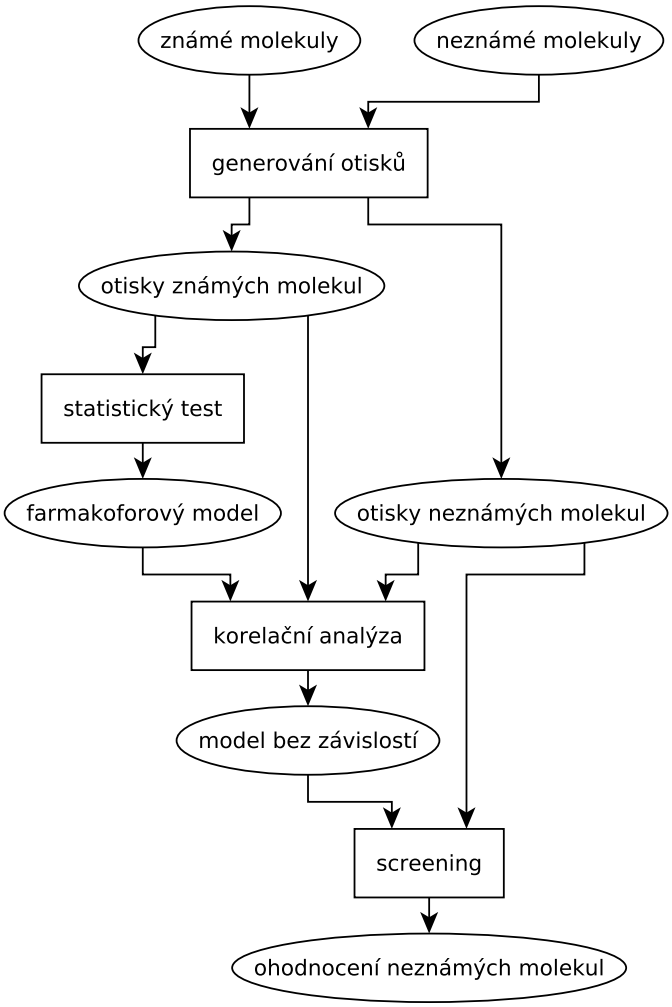
Přestože jsme ve svých experimentech zkoumali vliv několika parametrů na výkon metody, některé jsme ponechali pevné. Konkrétně jde o:

- farmakoforové vlastnosti
- koše pro přihrádkování vzdáleností
- hladinu statistického testu α

První dva parametry jsme ponechali stejné jako ve článku představujícím 2DPP metodu (Hoksza a Škoda, 2014), viz kapitola 2.1. Hladina testu byl parametr, který autoři zkoumali, konkrétně porovnávali $\alpha = 5\,\%$ a $\alpha = 1\,\%$. Na základě jejich výsledků jsme se rozhodli použít $\alpha = 5\,\%$, neboť při nižším α je metoda mnohem více ovlivněna počtem signifikantních farmakoforů. Jelikož při odstraňování závislostí počet signifikantních farmakoforů snižujeme, zdálo se nám vhodnější použít vyšší α, které není počtem farmakoforů tolik ovlivněné.

5.2.3 Porovnání s ostatními metodami

Pro porovnání našich výsledků s běžně používanými metodami jsme vybrali čtyři druhy molekulových otisků popsaných v kap. 1.2 – MACCS keys, AP, TT a ECFP4. Pro screening jsme použili Tanimotův koeficient a max-fusion. Screening byl proveden na stejně rozdělených datech jako v našem měření (tj. stejné složení trénovací a testovací sady, jen neaktivní molekuly z trénovací sady nebyly využity) a byl proveden pomocí nástroje LBVS Environment (Škoda, 2016).



Obrázek 5.2: Schéma průběhu 2DPP metody s odstraňováním závislostí.

5.3 Korelační analýza

Nejdříve ukážeme, jaký vliv má odstranění závislostí mezi signifikantními farmakofory na výkon metody. Průběh metody jsme upravili tak, že po vytvoření farmakoforového modelu jsme provedli korelační analýzu (viz kap. 4), pomocí které jsme odstranili závislosti mezi signifikantními farmakofory. Takto upravený farmakoforový model jsme použili ke screeningu (obrázek 5.2).

Pro výpočet korelačních koeficientů pro výskyty farmakoforů (skrze které se určí závislé farmakofory) jsme se rozhodli použít *celý dataset*. Důvodem pro to je, že použití pouze části datasetu by mohlo způsobit identifikování vztahů, které nejsou v celém datasetu, a tedy odstranění důležitých farmakoforů. Vzhledem k tomu, že pro určení závislostí není třeba vědět, které molekuly jsou aktivní, byl by tento postup použitelný i v praxi.

Naší hypotézou bylo, že odstraněním závislostí mezi farmakofory se výkon metody zvýší. Porovnali jsme tedy původní metodu používající dvou- až tříbodové farmakofory s naší modifikací odstraňující závislosti. Celkem jsme použili tři různé hranice pro závislost (viz kap. 4.1). Výsledky jsou shrnuty v tabulce 5.1.

	bez	t = 0.8	t = 0.5	t = 0.3
466	0.65	0.66	0.66	0.68
548	0.76	0.75	0.74	0.72
600	0.69	0.70	0.69	0.68
644	0.56	0.58	0.57	0.54
652	0.65	0.66	0.67	0.68
689	0.58	0.62	0.63	0.64
692	0.54	0.55	0.55	0.55
712	0.64	0.67	0.69	0.69
713	0.66	0.64	0.65	0.63
733	0.58	0.57	0.59	0.58
737	0.77	0.77	0.79	0.77
810	0.73	0.73	0.72	0.72
832	0.83	0.78	0.83	0.79
846	0.74	0.74	0.73	0.73
852	0.77	0.79	0.78	0.78
858	0.71	0.72	0.71	0.69
859	0.60	0.58	0.59	0.59
	0.67	0.68	0.68	0.67

		bez	t = 0.8	t = 0.5	t = 0.3
0.80-0.85	ADA2A	0.67	0.69	0.69	0.67
	CDK2	0.72	0.72	0.71	0.70
	HDAC01	0.59	0.59	0.61	0.57
	PXR_Agonist	0.66	0.66	0.64	0.58
	5HT2B	0.56	0.55	0.54	0.54
	5HT2C	0.54	0.53	0.58	0.58
0.85-0.90	ACM1_Agonist	0.62	0.58	0.56	0.57
	ADA2B_Antagonist	0.69	0.70	0.69	0.68
	ADA2C_Antagonist	0.68	0.70	0.68	0.66
	CHK1	0.66	0.61	0.61	0.63
0.90-0.95	DRD1_Antagonist	0.59	0.54	0.54	0.54
	DRD2_Agonist	0.67	0.68	0.67	0.68
	LSHR_Antagonist	0.76	0.72	0.69	0.73
	OPRM_Agonist	0.73	0.74	0.71	0.68
	5HT1F_Agonist	0.68	0.70	0.68	0.70
0.98-1	DHFR	0.86	0.89	0.87	0.78
	MTR1A_Agonist	0.55	0.61	0.60	0.58
	MTR1B_Agonist	0.54	0.61	0.67	0.56
	P38	0.69	0.73	0.70	0.74
	V2R_Antagonist	0.72	0.69	0.70	0.58
		0.66	0.66	0.65	0.64

Tabulka 5.1: Porovnání metody bez odstraňování závislostí (sloupec „bez“) a metody s odstraňováním závislostí pro různé hranice závislosti t. Nalevo jsou výsledky pro MUV datasety, napravo pro kompilaci různě obtížných datasetů (sloupec nejvíce nalevo je průměrně dosažené AUC na daných datasetech). Poslední řádek je průměrné AUC přes všechny datasety.

	bez	t = 0.8	t = 0.5	t = 0.3
466	0.71	0.71	0.72	0.75
548	0.74	0.75	0.75	0.74
600	0.70	0.71	0.74	0.74
644	0.55	0.55	0.59	0.59
652	0.67	0.68	0.70	0.72
689	0.64	0.65	0.66	0.67
692	0.54	0.56	0.60	0.61
712	0.68	0.70	0.72	0.74
713	0.67	0.67	0.67	0.66
733	0.57	0.60	0.61	0.60
737	0.78	0.78	0.80	0.80
810	0.72	0.72	0.71	0.71
832	0.81	0.81	0.83	0.83
846	0.72	0.72	0.75	0.76
852	0.78	0.79	0.81	0.83
858	0.70	0.70	0.72	0.72
859	0.58	0.58	0.58	0.59
	0.68	0.69	0.70	0.71

		bez	t = 0.8	t = 0.5	t = 0.3
0.80-0.85	ADA2A	0.65	0.62	0.56	0.54
	CDK2	0.75	0.76	0.77	0.78
	HDAC01	0.59	0.59	0.59	0.59
	PXR_Agonist	0.68	0.66	0.62	0.59
	5HT2B	0.57	0.58	0.56	0.55
	5HT2C	0.58	0.57	0.60	0.61
0.85-0.90	ACM1_Agonist	0.63	0.61	0.57	0.56
	ADA2B_Antagonist	0.66	0.63	0.59	0.54
	ADA2C_Antagonist	0.66	0.64	0.60	0.56
	CHK1	0.63	0.64	0.66	0.65
0.90-0.95	DRD1_Antagonist	0.55	0.61	0.66	0.67
	DRD2_Agonist	0.60	0.54	0.59	0.63
	LSHR_Antagonist	0.72	0.74	0.78	0.80
	OPRM_Agonist	0.61	0.54	0.55	0.56
	5HT1F_Agonist	0.74	0.75	0.78	0.77
0.98-1	DHFR	0.89	0.88	0.85	0.80
	MTR1A_Agonist	0.56	0.64	0.67	0.61
	MTR1B_Agonist	0.55	0.65	0.69	0.66
	P38	0.68	0.66	0.70	0.69
	V2R_Antagonist	0.67	0.61	0.51	0.57
		0.65	0.65	0.65	0.64

Tabulka 5.2: Porovnání metody bez odstraňování závislostí (sloupec „bez“) a metody s odstraňováním závislostí pro dvou- až pětibodové farmakofory a tři hodnoty hranice závislosti t. Nalevo jsou výsledky pro MUV datasety, napravo pro kompilaci různě obtížných datasetů. Poslední řádek je průměrné AUC přes všechny datasety.

Jak je vidět, tak odstranění závislostí nepřináší žádné výrazné zlepšení. Pro nízkou hranici závislosti dokonce může přinést i zhoršení (což ale není zas tak překvapivé). Možným vysvětlením by mohlo být snížení citlivosti metody způsobené snížením počtu signifikantních farmakoforů. Zkusili jsme tedy odstranění závislostí použít na dvou- až pětibodové farmakofory, které obsahují více farmakoforů (tabulka 5.2).

Ani při použití dvou- až pětibodových farmakoforů však odstraňování závislostí nijak výrazně výkon metody nezvyšuje. Na MUV datasetech je sice AUC při odstraňování závislostí vyšší než bez něj, ale jen o velmi málo. Naopak na kompilaci různě obtížných datasetů je odstraňování závislostí někdy i výrazně horší než původní metoda bez něj.

Jak je tedy vidět, odstranění závislostí nemá na metodu výrazný pozitivní vliv. Nicméně to, že se odstraňováním závislostí výsledky příliš nezměnily, naznačuje, že odstraněné farmakofory k identifikaci aktivních molekul moc nepřispěly. Naopak, snížení výkonu naznačuje, že odstraněné farmakofory byly ty, které opravdu rozlišují aktivní molekuly od neaktivních. Pro budoucí výzkum se tedy nabízí možnost použít korelační analýzu k zpřesnění farmakoforového modelu či k určení váhy signifikantních farmakoforů.

5.4 Vícebodové farmakofory

Jedním z parametrů 2D farmakoforové metody je množina farmakoforů, které jsou použity v otiscích. Vzhledem k použitým otiskům můžeme množinu farmakoforů ovlivnit určením počtu bodů ve farmakoforech. Zabývali jsme se tedy tím, jak je výkon metody závislý na tomto počtu. Vzhledem k tomu, že generování 2D farmakoforových otisků je náročné na výpočetní zdroje, byli jsme schopni prozkoumat nejvýše pětibodové farmakofory (i při použití výpočetních zdrojů Metacentra[1], které nám umožnily paralelní výpočty na stovkách strojů).

Prozkoumali jsme čtyři různé velikosti farmakoforů – dvoubodové, dvou- až tříbodové (použité v původní metodě), dvou- až čtyřbodové a dvou- až pětibodové. Výsledky jsou shrnuty v tabulce 5.3. Na obrázku 5.3 je potom průměrná hodnota AUC v závislosti na velikosti farmakoforů.

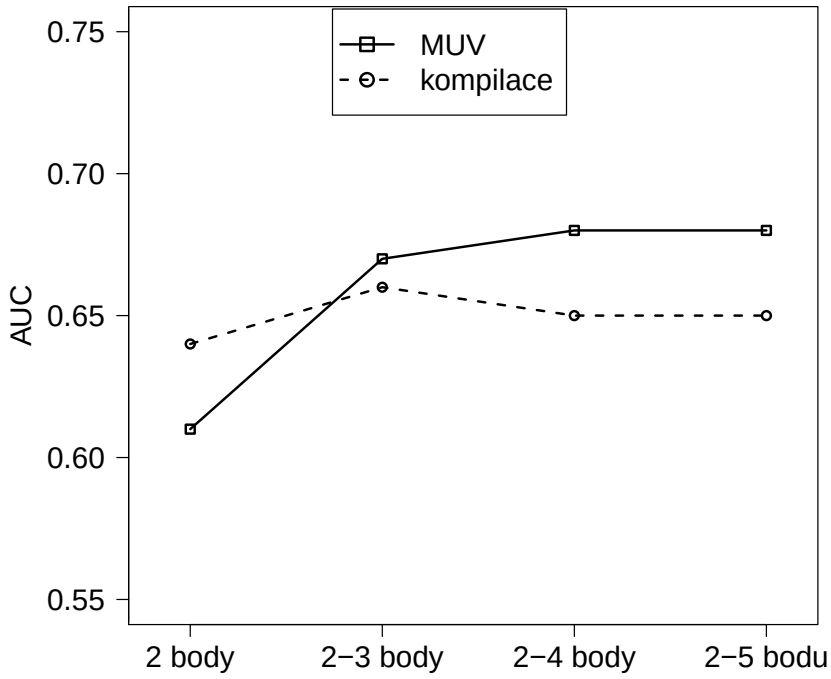
Jak je vidět, největší rozdíl je mezi dvoubodovými a dvou- až tříbodovými farmakofory, kde u MUV datasetů vzrostlo průměrné AUC téměř o deset procent. Oproti tomu rozdíl mezi (2-3)-bodovými a (2-4)-bodovými, nebo (2-4)-bodovými a (2-5)-bodovými je mnohem menší. Dokonce, v případě kompilace datasetů, je průměrné AUC (2-5)-bodových farmakoforů lehce menší než (2-3)-bodových. To odpovídá tomu, že největší rozlišovací schopnost mají tříbodové farmakofory i přesto, že při použití více bodů je signifikantních farmakoforů mnohem více (viz tab. 5.4). To by mohlo být způsobeno tím, že počet všech N-bodových farmakoforů roste velice rychle s hodnotou N a výskyty farmakoforů jsou tedy více „rozprostřeny“. Vyskytuje-li se nějaký farmakofor jen v malém počtu molekul, pak jeho výskyt v aktivní molekule má mnohem větší váhu pro statistický test kvůli nepoměru v počtu aktivních a neaktivních molekul. To by mohlo způsobovat „šum“ – farmakofory, které jsou signifikantní jen proto, že se vyskytují velmi málo, ale neodlišují aktivní molekuly od neaktivních.

[1] http://metavo.metacentrum.cz/

	2 body	2-3 body	2-4 body	2-5 bodu
466	0.56	0.65	0.70	0.71
548	0.69	0.76	0.76	0.74
600	0.62	0.69	0.71	0.70
644	0.53	0.56	0.56	0.55
652	0.62	0.65	0.67	0.67
689	0.54	0.58	0.62	0.64
692	0.49	0.54	0.56	0.54
712	0.57	0.64	0.66	0.68
713	0.60	0.66	0.68	0.67
733	0.50	0.58	0.58	0.57
737	0.76	0.77	0.78	0.78
810	0.58	0.73	0.73	0.72
832	0.84	0.83	0.81	0.81
846	0.70	0.74	0.74	0.72
852	0.74	0.77	0.77	0.78
858	0.67	0.71	0.70	0.70
859	0.43	0.60	0.60	0.58
	0.61	0.67	0.68	0.68

		2 body	2-3 body	2-4 body	2-5 bodu
0.80-0.85	ADA2A	0.65	0.67	0.66	0.65
	CDK2	0.68	0.72	0.74	0.75
	HDAC01	0.59	0.59	0.59	0.59
	PXR_Agonist	0.65	0.66	0.67	0.68
	5HT2B	0.54	0.56	0.56	0.57
	5HT2C	0.51	0.54	0.55	0.58
0.85-0.90	ACM1_Agonist	0.64	0.62	0.63	0.63
	ADA2B_Antagonis	0.65	0.69	0.67	0.66
	ADA2C_Antagonis	0.66	0.68	0.67	0.66
	CHK1	0.68	0.66	0.59	0.63
0.90-0.95	DRD1_Antagonist	0.65	0.59	0.55	0.55
	DRD2_Agonist	0.62	0.67	0.62	0.60
	LSHR_Antagonist	0.77	0.76	0.71	0.72
	OPRM_Agonist	0.68	0.73	0.68	0.61
	5HT1F_Agonist	0.57	0.68	0.72	0.74
0.98-1	DHFR	0.72	0.86	0.88	0.89
	MTR1A_Agonist	0.53	0.55	0.56	0.56
	MTR1B_Agonist	0.58	0.54	0.54	0.55
	P38	0.71	0.69	0.71	0.68
	V2R_Antagonist	0.72	0.72	0.70	0.67
		0.64	0.66	0.65	0.65

Tabulka 5.3: Porovnání výsledků metody pro různé velikosti farmakoforů.



Obrázek 5.3: Průměrný výkon metody v závislosti na počtu bodů ve farmakoforech.

	počet signifikantních farmakoforů	z toho v nejvýše 100 molekulách
2 body	1,3	0,5 %
2-3 body	20,6	3,6 %
2-4 body	189,4	7,5 %
2-5 bodů	1 183,5	12,2 %

Tabulka 5.4: Průměrný počet signifikantních farmakoforů a průměrný počet signifikantních farmakoforů vyskytujících se v nejvýše 100 molekulách datasetu.

	2DPP	AP	ECFP4	MACCS	TT
466	0.65	0.66	0.59	0.57	0.66
548	0.76	0.87	0.83	0.63	0.8
600	0.69	0.75	0.76	0.65	0.77
644	0.56	0.84	0.8	0.7	0.84
652	0.65	0.71	0.68	0.49	0.7
689	0.58	0.68	0.67	0.56	0.81
692	0.54	0.6	0.54	0.53	0.6
712	0.64	0.74	0.7	0.62	0.77
713	0.66	0.69	0.69	0.5	0.67
733	0.58	0.71	0.64	0.53	0.68
737	0.77	0.77	0.74	0.7	0.7
810	0.73	0.73	0.67	0.65	0.79
832	0.83	0.86	0.84	0.85	0.88
846	0.74	0.9	0.89	0.81	0.9
852	0.77	0.86	0.8	0.81	0.81
858	0.71	0.67	0.64	0.6	0.68
859	0.60	0.55	0.58	0.61	0.56
	0.67	0.74	0.71	0.64	0.74

		2DPP	AP	ECFP4	MACCS	TT
0.80-0.85	ADA2A	0.67	0.84	0.83	0.86	0.83
	CDK2	0.72	0.83	0.79	0.8	0.86
	HDAC01	0.59	0.81	0.83	0.76	0.83
	PXR_Agonist	0.66	0.83	0.85	0.87	0.86
	5HT2B	0.56	0.78	0.8	0.82	0.82
	5HT2C	0.54	0.85	0.85	0.85	0.86
0.85-0.90	ACM1_Agonist	0.62	0.87	0.89	0.9	0.85
	ADA2B_Antagonist	0.69	0.88	0.88	0.89	0.86
	ADA2C_Antagonist	0.68	0.87	0.88	0.89	0.87
	CHK1	0.66	0.91	0.9	0.86	0.96
0.90-0.95	DRD1_Antagonist	0.59	0.94	0.94	0.95	0.94
	DRD2_Agonist	0.67	0.95	0.95	0.96	0.96
	LSHR_Antagonist	0.76	0.93	0.93	0.92	0.96
	OPRM_Agonist	0.73	0.95	0.96	0.93	0.97
	5HT1F_Agonist	0.68	0.94	0.94	0.95	0.94
0.98-1	DHFR	0.86	0.98	0.99	1	1
	MTR1A_Agonist	0.55	0.99	0.99	0.98	0.99
	MTR1B_Agonist	0.54	0.99	0.99	0.99	0.99
	P38	0.69	0.98	0.99	0.98	0.99
	V2R_Antagonist	0.72	0.98	0.98	0.98	0.99
		0.66	0.91	0.91	0.91	0.92

Tabulka 5.5: Porovnání výsledků s ostatními metodami. 2DPP je původní metoda (s dvou- až tříbodovými farmakofory).

5.5 Kombinace hodnotících metod

Při porovnání výsledků 2D farmakoforové metody s jinými metodami založenými na otiscích (tabulka 5.5) se ukázalo, že na kompilaci datasetů si 2DPP metoda vede mnohem hůře. A to i přesto, že by mělo jít o „jednoduché“ datasety, přičemž na MUV datasetech, které jsou „těžké“, si vede poměrně dobře.

Vysvětlení by mohlo být ve struktuře dat. MUV datasety byly navrženy tak, že aktivní molekuly jsou rovnoměrně rozmístěny v chemickém prostoru a každá aktivní je rovnoměrně obklopena neaktivními molekulami (Rohrer a Baumann, 2009). MUV datasety tedy neobsahují shluky aktivních molekul. To znevýhodňuje screeningové metody, které mají ohodnocení založené na nejmenší vzdálenosti ke známým aktivním molekulám (jako je max-fusion s Tanimotovým koeficientem). Na takových datasetech by mělo být výhodnější použití metod založených na identifikaci vlastností společných pro všechny aktivní molekuly (jako je 2D farmakoforová metoda).

Naopak kompilace datasetů byla navržena tak, aby na ní měly dobrý výkon právě metody založené na nejmenší vzdálenosti ke známým aktivním molekulám. Je tedy pravděpodobné, že tyto datasety budou obsahovat shluky aktivních molekul. Pokud budou známé molekuly vybrány z různých shluků, bude mít většina aktivní molekul ve své blízkosti nějakou známou molekulu. Metody používající jako skóre vzdálenost ke známým aktivním molekulám budou mít proto velmi dobré výsledky. Naopak model vytvořený 2DPP metodou je složen z toho, co mají známé molekuly společné, a je tak známým molekulám dál (známé aktivní molekuly při použití max-fusion dostanou nejvyšší možné skóre, při použití 2DPP metody tomu tak není).

Rozhodli jsme se tedy využít jak vzdálenost ke známým aktivním molekulám, tak informace obsažené ve farmakoforovém modelu. Místo skóre používaného v původní 2DPP metodě jsme použili max-fusion spolu s Tanimotovým koeficientem, ale každému bitu v otisku jsme přiřadili i váhu v závislosti na modelu. Přesněji, skóre molekuly M pro daný farmakoforový model P a množinu známých aktivních molekul $\mathcal{A}$ se spočítá jako

$$S(M,P) = \max_{A \in \mathcal{A}} \left(T(M,A,P) \right),$$

kde $T(M,A,P)$ je vážený Tanimotův koeficient:

$$T(M,A,P) = \frac{\sum_{i=1}^{l} M[i] \cdot A[i] \cdot w(i,P)}{\sum_{i=1}^{l} \left(M[i] + A[i] - M[i] \cdot A[i] \right) \cdot w(i,P)},$$

přičemž l je počet bitů farmakoforového otisku, $M[i]$, resp. $A[i]$, je i-tý bit otisku molekuly M, resp. A, a $w(i,P)$ je váha i-tého bitu.

Vyzkoušeli jsme tři různé váhové funkce:

- TAN1: $w(i,P) = 1$ pro každé i (klasický Tan. koef.)
- TAN2: $w(i,P) = 1 + P[i]$ (farmakofory v modelu mají dvojnásobnou váhu)
- TAN10: $w(i,P) = 1 + 9 \cdot P[i]$ (farm. v modelu mají desetinásobnou váhu).

Výsledky jsou shrnuty v tabulce 5.6

Použití max-fusion s Tanimotovým koeficientem na MUV datasetech vedlo ke snížení výkonu, naopak na kompilaci datasetů došlo k velmi výraznému zlepšení, což bylo to, co jsme očekávali. Zvýšení váhy farmakoforů obsažených v modelu vedlo pouze k drobnému zlepšení, v několika případech dokonce došlo k drobnému zhoršení.

	2DPP	TAN1	TAN2	TAN10
466	0.65	0.55	0.58	0.59
548	0.76	0.75	0.77	0.80
600	0.69	0.57	0.58	0.59
644	0.56	0.67	0.67	0.66
652	0.65	0.57	0.58	0.57
689	0.58	0.54	0.54	0.58
692	0.54	0.58	0.59	0.59
712	0.64	0.53	0.52	0.54
713	0.66	0.56	0.56	0.57
733	0.58	0.60	0.60	0.58
737	0.77	0.69	0.70	0.71
810	0.73	0.54	0.55	0.60
832	0.83	0.82	0.84	0.86
846	0.74	0.70	0.68	0.71
852	0.77	0.77	0.78	0.78
858	0.71	0.54	0.54	0.55
859	0.60	0.58	0.57	0.57
	0.67	0.62	0.63	0.64

		2DPP	TAN1	TAN2	TAN10
0.80-0.85	ADA2A	0.67	0.77	0.77	0.74
	CDK2	0.72	0.76	0.77	0.80
	HDAC01	0.59	0.63	0.64	0.62
	PXR_Agonist	0.66	0.82	0.81	0.79
	5HT2B	0.56	0.74	0.74	0.71
	5HT2C	0.54	0.77	0.78	0.78
0.85-0.90	ACM1_Agonist	0.62	0.79	0.79	0.76
	ADA2B_Antagonist	0.69	0.79	0.78	0.74
	ADA2C_Antagonist	0.68	0.74	0.75	0.74
	CHK1	0.66	0.79	0.80	0.81
0.90-0.95	DRD1_Antagonist	0.59	0.85	0.86	0.87
	DRD2_Agonist	0.67	0.84	0.86	0.86
	LSHR_Antagonist	0.76	0.86	0.88	0.90
	OPRM_Agonist	0.73	0.81	0.82	0.79
	5HT1F_Agonist	0.68	0.86	0.87	0.87
0.98-1	DHFR	0.86	0.98	0.98	0.99
	MTR1A_Agonist	0.55	0.96	0.97	0.98
	MTR1B_Agonist	0.54	0.95	0.97	0.97
	P38	0.69	0.96	0.96	0.95
	V2R_Antagonist	0.72	0.91	0.92	0.92
		0.66	0.81	0.82	0.81

Tabulka 5.6: Porovnání původní metody s metodami kombinujícími podobnostní koeficient s farmakoforovým modelem.

Závěr

Hlavními cíli této práce bylo prozkoumat vliv použití vícebodových farmakoforů a korelační analýzy na 2D farmakoforovou metodu. Pro použití vícebodových farmakoforů bylo nutné zjistit, proč RDKit (použitý k tvorbě 2D farmakoforových otisků molekul) podporuje pouze dvou- a tříbodové farmakofory a následně naimplementovat příslušné rozšíření. To se nám i přes absenci kvalitní dokumentace RDKitu povedlo. Navíc se nám podařilo tvorbu otisku zefektivnit. Dále jsme navrhli a naimplementovali algoritmus pro odstraňování korelací (závislostí) mezi farmakofory.

Experimentální výsledky ukázali, že použití tříbodových farmakoforů přináší oproti těm dvoubodovým výrazné zlepšení. Nicméně, další zvýšení velikosti farmakoforů k zlepšení nevede. Domníváme se, že důvodem jsou farmakofory, které se vyskytují jen v malém množství molekul. Ani korelační analýza nepřinesla žádné výrazné zlepšení. Poukázali jsme však na možnost využití korelační analýzy ke zpřesnění farmakoforového modelu. Nakonec, na základě špatných výsledků metody na „jednoduchých“ datasetech, jsme navrhli použití farmakoforového modelu k vylepšení metod používající podobnost otisků. Tím jsme sice dosáhli výrazného zlepšení na „jednoduchých“ datasetech, ale za cenu zhoršení na „těžkých“ datasetech.

V práci jsme navrhli několik dalších možností, které mají potenciál 2D farmakoforovou metodu zlepšit. Jednak je zde možnost využít korelační analýzu k lepší identifikaci farmakoforů, které oddělují aktivní molekuly od neaktivních. To by mohlo být užitečné obzvláště u více než tříbodových farmakoforů, jejichž výkon může být limitován právě „šumem“ mezi signifikantními farmakofory. Dále je zde kombinace farmakoforového modelu s podobností otisků, což je směr, který, jak se domníváme, nabízí velký prostor pro možná zlepšení metody. Také je zde možnost zlepšit metodu po „softwarové stránce“, například efektivní implementací tvorby farmakoforových otisků, či implementací uživatelsky přívětivého rozhraní pro celou metodu.

Seznam použité literatury

Brint, A. T. a Willett, P. (1987). Pharmacophoric pattern matching in files of 3D chemical structures: comparison of geometric searching algorithms. *Journal of Molecular Graphics*, **5**(1), 49–56.

Carhart, R. E., Smith, D. H. a Venkataraghavan, R. (1985). Atom pairs as molecular features in structure-activity studies: definition and applications. *Journal of Chemical Information and Computer Sciences*, **25**(2), 64–73.

Dror, O., Shulman-Peleg, A., Nussinov, R. a Wolfson, H. J. (2004). Predicting molecular interactions in silico: I. a guide to pharmacophore identification and its applications to drug design. *Current medicinal chemistry*, **11** (1), 71–90.

Ehrlich, P. (1909). Über den jetzigen stand der chemotherapie. *Berichte der deutschen chemischen Gesellschaft*, **42**(1), 17–47.

Gobbi, A. a Poppinger, D. (1998). Genetic optimization of combinatorial libraries. *Biotechnology and bioengineering*, **61**(1), 47–54.

Hoksza, D. a Škoda, P. (2014). 2D Pharmacophore query generation. In Basu, M., Pan, Y. a Wang, J., editors, *Bioinformatics Research and Applications*, volume 8492 of *Lecture Notes in Computer Science*, pages 289–300. Springer International Publishing. ISBN 978-3-319-08170-0. doi: 10.1007/978-3-319-08171-7˙26. URL `http://dx.doi.org/10.1007/978-3-319-08171-7_26`.

Hoksza, D. a Škoda, P. (In-press). Using bayesian modeling on molecular fragments features for virtual screening. In *IEEE International Conference on Computational Intelligence in Bioinformatics and Computational Biology 2016*.

Ihaka, R. a Gentleman, R. (1996). R: a language for data analysis and graphics. *Journal of computational and graphical statistics*, **5**(3), 299–314.

Jones, E., Oliphant, T., Peterson, P. a kol. (2001). Open source scientific tools for python.

Kumar, R. a Suresh, M. (2013). Pharmacophore mapping based inhibitor selection and molecular interaction studies for identification of potential drugs on calcium activated potassium channel blockers, tamulotoxin. *Pharmacognosy Magazine*, **9**(34), 89–95. doi: 10.4103/0973-1296.111239. URL `http://www.phcog.com/article.asp?issn=0973-1296;year=2013;volume=9;issue=34;spage=89;epage=95;aulast=Kumar;t=6`.

Landrum, G. (2006). RDKit: Open-source cheminformatics [online] . http://www.rdkit.org. Accessed: 2016-5-13.

Landrum, G. (2015a). RDKit documentation. *Release 2015.09.1*, pages 1–79.

Landrum, G. (2015b). RDKit: Open-source cheminformatics (release 2015.09.1) [source codes]. Module rdkit.Chem.Pharm2D. https://github.com/rdkit/rdkit/tree/master/rdkit/Chem/Pharm2D . Accessed: 2016-5-23.

Leach, A. R. a Gillet, V. J. (2007). *An introduction to chemoinformatics*. Springer Science & Business Media.

Lee Rodgers, J. a Nicewander, W. A. (1988). Thirteen ways to look at the correlation coefficient. *The American Statistician*, **42**(1), 59–66.

McGregor, M. J. a Pallai, P. V. (1997). Clustering of large databases of compounds: Using the mdl "keys" as structural descriptors. *Journal of chemical information and computer sciences*, **37**(3), 443–448.

Morgan, H. (1965). The generation of a unique machine description for chemical structures-a technique developed at chemical abstracts service. *Journal of Chemical Documentation*, **5**(2), 107–113.

Nilakantan, R., Bauman, N., Dixon, J. S. a Venkataraghavan, R. (1987). Topological torsion: a new molecular descriptor for sar applications. comparison with other descriptors. *Journal of Chemical Information and Computer Sciences*, **27**(2), 82–85.

Robin, X., Turck, N., Hainard, A., Tiberti, N., Lisacek, F., Sanchez, J.-C. a Müller, M. (2011). pROC: an open-source package for R and S+ to analyze and compare ROC curves. *BMC Bioinformatics*, **12**, 77.

Rogers, D. a Hahn, M. (2010). Extended-connectivity fingerprints. *Journal of chemical information and modeling*, **50**(5), 742–754.

Rohrer, S. G. a Baumann, K. (2009). Maximum unbiased validation (MUV) data sets for virtual screening based on PubChem bioactivity data. *Journal of chemical information and modeling*, **49**(2), 169–184.

Steinbeck, C., Han, Y., Kuhn, S., Horlacher, O., Luttmann, E. a Willighagen, E. (2003). The chemistry development kit (CDK): An open-source java library for chemo-and bioinformatics. *Journal of chemical information and computer sciences*, **43**(2), 493–500.

Sun, H. (2008). Pharmacophore-based virtual screening. *Current medicinal chemistry*, **15**(10), 1018–1024.

Sun, H., Greeley, D. N., Chu, X.-J., Cheung, A., Danho, W., Swistok, J., Wang, Y., Zhao, C., Chen, L. a Fry, D. C. (2004). A predictive pharmacophore model of human melanocortin-4 receptor as derived from the solution structures of cyclic peptides. *Bioorganic & medicinal chemistry*, **12** (10), 2671–2677.

Taboureau, O., Baell, J. B., Fernández-Recio, J. a Villoutreix, B. O. (2012). Established and emerging trends in computational drug discovery in the structural genomics era. *Chemistry & biology*, **19**(1), 29–41.

Triballeau, N., Acher, F., Brabet, I., Pin, J.-P. a Bertrand, H.-O. (2005). Virtual screening workflow development guided by the "receiver operating characteristic" curve approach. application to high-throughput docking on metabotropic glutamate receptor subtype 4. *Journal of medicinal chemistry*, **48**(7), 2534–2547.

Wermuth, C., Ganellin, C., Lindberg, P. a Mitscher, L. (1998). Glossary of terms used in medicinal chemistry (IUPAC recommendations 1998). *Pure and Applied Chemistry*, **70**(5), 1129–1143.

Willett, P. (2006). Similarity-based virtual screening using 2D fingerprints. *Drug discovery today*, **11**(23), 1046–1053.

Willett, P. (2011). Similarity searching using 2D structural fingerprints. *Chemoinformatics and computational chemical biology*, pages 133–158.

Škoda, P. (2016). LBVS Environment [online] . https://github.com/skodape/lbvs-environment. Accessed: 2016-6-15.

Seznam obrázků

Seznam tabulek

Přílohy

Příloha 1 – Naměřená data

Soubor `data.ods` obsahuje všechna naměřená data ve formátu OpenDocument. Dokument obsahuje dva listy, jeden pro MUV datasety a druhý pro kompilaci datasetů. Data jsou rozdělena do dvou tabulek. První obsahuje AUC pro metody bez odstraňování závislostí, druhá naopak s odstraňováním závislostí. Data jsou rozdělena podle hranice závislosti (jen u druhé tabulky), pak podle počtu bodů ve farmakoforech a nakonec podle metody pomocí které se určovalo skóre molekul při screeningu. Metoda označená 2DPP používá skóre S_{pp} definované v kap. 2.3, metoda označená COMB používá skóre S_{comb} definované tamtéž. Metody označené TAN1, TAN2 a TAN10 používají odpovídající skóre definované v kap. 5.5.

Příloha 2 – Modifikace RDKitu

Složka `rdkit/` obsahuje soubory `Generate.py`, `SigFactory.py` a `Utils.py`. Jedná se o upravené zdrojové kódy částí modulu `rdkit.Chem.Pharm2D` implementující naše modifikace RDKitu. Pro jejich použití v RDKitu stačí těmito soubory přepsat odpovídající soubory z modulu `rdkit.Chem.Pharm2D`.

Příloha 3 – Implementace metody

Složka `screening/` obsahuje data a skripty implementující všechny části metod popsaných v této práci. Ve složce `datasets/` jsou obsaženy všechny použité datasety. Tato složka obsahuje pět složek – jednu pro MUV datasety a jednu pro každou ze čtyř skupin datasetů z kompilace. Ty obsahují jednak soubory s molekulami (ve formátu SDF, zkomprimované programem `gzip`) a jednak testovací a trénovací sady molekul.

Všechny skripty čtou parametry z konfiguračního souboru ve formátu INI. Jeho jméno buď dostanou jako poslední argument, nebo použijí soubor se jménem `config` ze složky ve které jsou spuštěny. Ukázkový konfigurační soubor je přiložen. Složky, ze kterých skripty čtou data a do kterých zapisují výsledky, jsou specifikovány v konfiguračním souboru. Ukázková adresářová struktura pro vstup a výstup je ve složce `results/`.

Prvním skriptem v celé metodě je `gen_FPs.py`. Ten vygeneruje farmakoforové otisky všech molekul v zadaných datasetech a zapíše je do složky `results/FPs/`. Alternativně je možné generovat otisky po částech (což je vhodné pro čtyř- a pětibodové otisky). Nejdříve je třeba skriptem `split_mol_files.py` rozdělit soubory s molekulami na menší části. Pro každou takovou část je pak možné vygenerovat otisky skriptem `gen_FPs_parts.py`, který jako argumenty bere typ molekul (`actives` nebo `decoys`), název datasetu a číslo části (ignoruje datasety zadané v kofiguračním souboru). Výsledné soubory spojí skript `cat_FPs_files.py`.

Druhým skriptem v pořadí je `create_model.py`, který vytvoří farmakoforový model ze zadaných molekul. Z modelu pak lze odstranit závislé farmakofory pomocí skriptu `corr_matrix.py`, který vytvoří korelační matici výskytů signifikantních farmakoforů, a `remove_correlated.py`, který provede samotné odstranění závislostí.

Je-li vytvořen model (ať už se závislostmi nebo bez), je možno provést samotný screening pomocí skriptu `screening.py`. Ten zapíše výsledky do adresáře `results/screen/` (pro každý dataset jeden soubor za každou testovací sadu molekul). Výsledky je možné zpracovat skriptem `analyse_results.r`, který jako argumenty dostane právě soubory s výsledky screeningu, pro každý spočítá a vypíše AUC a nakonec vypíše průměrné AUC.

Všechny skripty jsou v jazyce Python (verze 2.7), s výjimkou skriptu `analyse_results.r`, který je v jazce R. Tento skript také vyžaduje knihovnu `pROC` (Robin a kol., 2011).