

# Simulování života mikroorganismů v modelovém mikrosvětě

Filip Majer  
Gymnázium Jana Keplera

<https://github.com/nat-int/mikrosim>

*dokumentace je nedokončená*

## Obsah

<b>1. Úvod .....</b>	<b>2</b>
1.1. Cíle .....	2
1.2. Související práce .....	2
<b>2. Model .....</b>	<b>2</b>
2.1. Prostředí .....	2
2.2. Proteiny .....	3
2.2.1. Allosterická místa .....	4
2.2.2. Reakce .....	5
2.2.3. Genom .....	6
2.2.4. Speciální funkce .....	6
2.3. Buňky .....	6
<b>3. Implementace .....</b>	<b>6</b>
3.1. Použité technologie .....	6
<b>4. Návod k použití .....</b>	<b>6</b>
4.1. Kompilace .....	6
4.2. Návod k aplikaci .....	7
4.2.1. okno „Controls“ .....	8
4.2.2. okno „mikrosim“ .....	9
4.2.3. okno „effect blocks“ .....	10
4.2.4. okno „extra info“ .....	10
4.2.5. okno „cell list“ .....	11
4.2.6. okno „protein view“ .....	11
4.2.7. okno „cell view“ .....	11
4.2.8. zobrazení simulace .....	11
<b>5. Závěr .....</b>	<b>11</b>

# 1. Úvod

Život je plný komplexních systémů, ale pro jeho zkoumání je někdy užitečné mít zjednodušený model, který jsme schopni plně propočítat. Naše schopnosti počítání posunuly vpřed počítače, kterým dnes už můžeme dávat i poměrně komplikované modely.

## 1.1. Cíle

Cílem tohoto projektu je vybrat a naimplementovat do grafické aplikace model mikroorganismů, takový, aby nebyl moc daleko od středoškolského modelu toho, jak mikroorganismy fungují, ale zároveň byl dostatečně jednoduchý, aby podle něj šlo simulovat mnoho organismů i na běžně rozšířených počítačích.

## 1.2. Související práce

Existuje hodně modelů i jejich simulátorů zjednodušeného života. Velká část z nich je udělaná tak, že se připraví několik typů buněk, kde má každá své specifické chování, ze kterých se podle poskládá organismus. Ten se pak může replikovat do stejné nebo podobné konfigurace. Jedním takovým simulátorem je Biogenesis (<https://biogenesis.sourceforge.net/>).

Jiná velká skupina modelů simuluje organismy jako objekty, které podle nějakých pravidel pozorují okolí a podle toho provádí různé akce, typicky s využitím neuronových sítí. Příkladem pro tuto skupinu je The bibites (<https://thebibites.itch.io/the-bibites>).

Oba tyto přístupy tak modelují spíše větší organismy než mikroorganismy.

O něco univerzálnější model má projekt ALIEN (<https://alien-project.org/>), který je založený na simulaci částic, které mohou tvořit vazby a předávat si informace. ALIEN se snaží o jednoduchý model a z něj vyvstávající chování, a tak je v něm život není tak blízko realitě, jako bychom zde chtěli. Kromě toho je napsaný v CUDA, takže vyžaduje počítač s grafickou kartou od společnosti Nvidia.

# 2. Model

Model je ve dvou dimenzích, aby bylo možné ho přehledně vykreslovat a snáze implementovat.

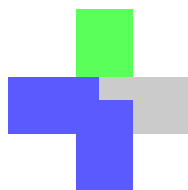
## 2.1. Prostředí

Mikroorganismy žijí v nějakém prostředí. Jako základ prostředí této simulace byla zvolena tekutina, protože se mnohým mikroorganismům žije dobře ve vodě. Standardní způsoby simulování tekutin jsou částicové, eulerovské a kombinované, ze kterých byl vybrán částicový model z tohoto simulátoru: <https://github.com/SebLague/Fluid-Sim/tree/Episode-01>, bez korekcí shlukování částic při rychlých ztrátách tlak, a to především proto, že je tento model poměrně snadný k implementaci, zároveň je pro částicové simulace snazší dosáhnout toho, aby se nevytvářela hmota z ničeho. Snadnost implementace tu je na úkor realismu, tekutina se chová jako něco mezi plynem, kapalinou a želé, ale stále to je dostatečně blízko reálné tekutině.

Kromě vody jsou v typickém prostředí, kde mikroorganismy žijí různé látky ze kterých získávají energii nebo stavební materiály. Aby se simulace vešla do paměti a zároveň lépe paralelizovala, jsou zde látky zjednodušené na seřazené čtveřici podčástí, které jsou červené,

## 2.1 Model — Prostředí

zelené, modré a šedé, ale shodné pod rotací, takže reprezentují čtyřcípou hvězdu s cípy obarvenými na jednu ze čtyř barev.



Obrázek 1: „látka“ složená ze zelené, šedé, modré a modré podčásti

Chování látek je pak inspirováno tím, jak se chová malá organická látka, která obsahuje jeden uhlík se čtyřmi skupinami, kde červená je trochu jako karboxylová skupina, zelená jako halogen (u zelené to zrovna není tak moc), modrá jako aminová skupina a šedá jako vodík. Ke každé látce je přiřazena energie (uložená v jejích vazbách), která je určena:

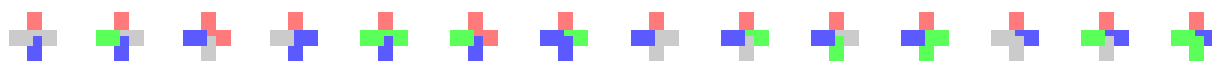
$$\sum_i^{\text{podčásti}} E_{z(b_i)} + c(b_i) \cdot (f(b_{\text{na jednu stranu od } i}) + f(b_{\text{na druhou stranu od } i}) + \varphi(b_{\text{naproti } i}))$$

kde  $b$  je barva na dané pozici, a  $E_z$ ,  $c$ ,  $f$  a  $\varphi$  jsou funkce  $[\text{barva}] \rightarrow \mathbb{Z}$ .  $E_z$  dává energii vazby mezi podčástmi a středem látky. Zbytek tohoto výpočtu je udělaný podle indukčního efektu.  $f$  určuje jeho sílu mezi vedlejšími pozicemi,  $\varphi$  mezi protějšími pozicemi a  $c$  sílu vlivu tohoto efektu na podčást. Energie látek jsou tak rozmanité a jen občas shodné (ale jsou shodné pro ekvivalentní enantiomery) a můžeme získávat zajímavé výsledky při počítání s jejich rozdíly.

Každá částice v sobě může mít libovolné množství každé látky. V průběhu simulace se látky rozpouští, tak, že při kroku, během kterého se daná látka má rozpustit, tak se z každé částice látka rovnoměrně rozloží do všech částic v okruhu se stejným poloměrem, jako je dosah jejich silových interakcí.

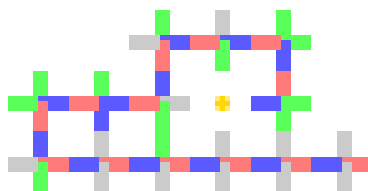
## 2.2. Proteiny

Funkce buněk provádí proteiny. Ty se skládají z aminokyselin, které se zřetězí a složí do tvaru, který jim umožní pracovat. Skládání proteinů je na tuto simulaci moc složité, a tak je zde model opět hodně zjednodušený. Z látek bylo vybráno několik jako aminokyseliny tvořící proteiny, které všechny obsahují červenou a modrou podčást:



Obrázek 2: látky vybrané jako proteinogenní

Ze seznamu těchto látek pak skládáme protein tak, že za červený konec předchozí látky přidáme další látku tak, aby navazovala modrým koncem, tak, jako se při translaci protein skládá od N-terminu a tvoří se peptidické vazby, akorát zde s pevně daným tvarem. Když má látka více červených nebo modrých konců, tak se zapojí dvojice podčástí naproti sobě.



Obrázek 3: ukázka proteinu (genom je aa19c6a80100e (více v sekci genom)), žlutá tečka značí aktivní místo

Ze struktury je určena stabilita proteinu - část, která vydrží krok simulace - takto:

$$\min(1 - 15\% \cdot e^{-s_c}, 99\%), s_c = \frac{n}{80} + \frac{n_e}{4}$$

kde  $n$  je počet míst, kde se protein překrývá sám se sebou a  $n_e$  je počet míst, kde se překrývá a zároveň má vedle daného místa volno. To dává způsoby k velkému zvětšování stability i malým jednodušším krokům, které stabilitu posunou o kousek dále, a reflektuje to, že zabalenější proteiny bývají stabilnější.

Jako aktivní místa jsou označeny ty vrcholy mřížky, ve které mohou být středy látek, které jsou ohraničeny ze všech čtyř stran právě jednou látkou proteinu (řetězec látek se tam nekříží). Do takového místa se může vázat látka, která má podčásti doplňkové k podčástem, které do něj směřují z ohraničujících látek. K červené podčásti je doplňková modrá a obráceně, k zelené části zelená a k šedé části šedá, přibližně na základě interpretace podčástí jako chemických skupin. Obrázek 3 obsahuje vyznačené místo, do kterého by se vázala látka [červená, šedá, šedá, zelená]. Aktivní místa mohou být maximálně 4, když jich je více, tak se preferují ty vlevo a poté nahoře.

Aktivní místa, která jsou maximálně 4 místa od sebe (ob 2 místa, opět přednostně zleva a shora), se zpárují a katalyzují reakci výměny podčástmi mezi látkami, těch dvou, které jsou nejbližší (s případnou preferencí vertikální výměny právě když je levé místo výše). Zbylá místa katalyzují reakci výměny vedlejších podčástí vázané látky, které jsou nejbližší středu (těžišti) aktivních míst (opět přednostně nalevo nahoře). Pro Obrázek 3 bude výsledek reakce [červená, šedá, zelená, šedá].

Tento způsob umožňuje převod pořadí látek na funkci proteinu, které souvisí se strukturou, a tak se při mutaci může funkce zachovat, mírně pozměnit i radikálně změnit s nezanedbatelnými pravděpodobnostmi a je implementačně i výpočetně poměrně jednoduchý. Také dokud nenastanou krajní případy (existuje ale významný krajní případ - jen jedno aktivní místo), tak je tento způsob invariantní pod rotací.

### 2.2.1. Allosterická místa

Funkce proteinů samotná může být aktivována i inhibována látkami. V tomto modelu jsou allosterická místa velmi podobná aktivním, jen mají v jednom směru prázdné místo, nebo překryté látky. Katalyzátory jsou pak 4 látky které se do daného místa váží stejným způsobem jako v aktivních místech, z podobných důvodů. Katalytický efekt je určen jako  $\sin(\text{vzdálenost allosterického od nejbližšího aktivního místa})$ . Sinus zde slouží jako docela

## 2.2 Model – Proteiny

hladká (s docela malou derivací) náhodná funkce, hladká aby malá změna ve struktuře mohla jen mírně změnit katalytický efekt.

Podle katalytického efektu se určuje celkový efekt katalyzátorů na protein:

$$K_k = \max \left( 0, 1 + \left( \sum_i^{\text{kat.}} f_i(c_k) \text{ když } f_i(c_k) > 0 \right) + \left( \min_i^{\text{kat.}} f_i(c_k) \text{ když } f_i(c_k) \leq 0 \right) \right)$$

kde  $c_k$  je koncentrace katalyzující látky v částici kde protein působí a

$$f_i(x) = x^2 + k_e - 1 \text{ pro } k_e > 0$$

$$f_i(x) = \text{lerp}(x^2 - 1, -x, \text{smoothstep}(-k_e)) \text{ pro } k_e \leq 0$$

kde  $k_e$  je katalytický efekt katalyzátoru.  $f_i$  je lineárně interpolovaná funkce vytvořená za krajních funkcí koncentrace, kde  $k_e = 1$  znamená, že katalyzátor reakci podporuje,  $k_e = 0$  znamená, že katalyzátor je k reakci potřeba a  $k_e = -1$  znamená, že katalyzátor reakci inhibuje. Pro záporná  $k_e$  je přidán smoothstep k interpolačnímu koeficientu, aby se urychlil přechod přes bod, kde  $f_i$  vychází přibližně jako záporná konstanta.

### 2.2.2. Reakce

Když protein působí v nějaké částici, katalyzuje v ní reakci. Předpokládá se, že reakce běží jen zanedbatelně bez pomoci proteinu, a tak protein spouští reakci. O reakci jsou známy reaktanty a produkty, takže lze získat i změnu entalpie:

$$\Delta H = \sum_i^{\text{produkty}} E(i) - \sum_i^{\text{reaktanty}} E(i)$$

kde  $E$  je již zmíněná energie látky. Když se zanedbá  $\Delta S \cdot T$ , tak z

$$\Delta G = \Delta H - \Delta S \cdot T \rightarrow \Delta G = \Delta H$$

$$K = e^{-\frac{\Delta G}{RT}} \rightarrow K = e^{-\frac{\Delta H}{RT}}$$

jde spočítat rovnovážnou konstantu reakce. K té by se měl blížit poměr součinů koncentrací produktů a reaktantů, s rychlostí určenou silou enzymatické katalýzy. Během kroku, kdy protein provádí danou reakci, se spočítá aktuální poměr ( $K''$ ) a během kroku se změní na

$$K' = \text{lerp}(K'', K, c_r \cdot K_k \cdot c_p)$$

kde  $c_r$  je volitelná konstanta určující rychlosti všech reakcí,  $K_k$  je efekt katalyzátorů a  $c_p$  je koncentrace proteinu. Nechť je  $\delta$  změna koncentrací v tomto kroku. Pak

$$K' = \frac{\prod_i^{\text{produkty}} (c_{pi} + \delta)}{\prod_i^{\text{reaktanty}} (c_{ri} - \delta)}$$

$$K' \cdot \left( \prod_i^{\text{reaktanty}} (c_{ri} - \delta) \right) - \prod_i^{\text{produkty}} (c_{pi} + \delta) = 0$$

což je po roznásobení rovnice maximálně čtvrtého stupně v  $\delta$  (protože maximum jsou 4 aktivní místa) a tak je řešitelná. Z řešení se vybere takové  $\delta$ , které je v absolutní hodnotě nejmenší a zároveň nedostane žádnou koncentraci do záporné hodnoty zmenšuje ho dle potřeby.

### 2.2.3. Genom

Buňky potřebují schopnost se replikovat, k čemuž jim slouží jejich genom, který slouží jako seznam všeho, z čeho buňka je. Zde je genom opět zjednodušený, protože musí být z „látek“ a měl by existovat způsob jeho interakce s proteiny, a je tvořen z látek [červená, zelená, šedá, zelená] a [modrá, zelená, šedá, zelená] vázaných do řady přes zelenou - takže tu jsou jen dvě možné báze. Kromě toho má genom implicitně určený začátek.

Ke genomu se může vázat jiný typ proteinů, transkripční faktory. Pokud má protein na okraji osově zarovnaného obdélníku obalujícího protein alespoň 7 po sobě jdoucích vystupujících červených nebo modrých podčástí, tak se z něj stává transkripční faktor a ovlivňuje projevy genomu v místě, kde se na genom váže, což je na genomu v místě, kde obsahuje komplementární sekvenci bází (k červené se váže modrá a obráceně). Jsou dva možné směry orientace genomu oproti transkripčnímu faktoru, ze kterého je vybrán ten, který je shodný s tím, že je genom poskládaný shora dolů s šedými podčástmi napravo (orientaci proteinu určuje jeho chiralita).

Navázaný transkripční faktor blokuje v daném místě transkripci, z důvodu snadné implementace působí v místě nejbližší začátku genomu. Pokud však na okraji obsahuje řadu alespoň tří zelených podčástí (ve stejném smyslu jako v části, která se váže na genom), tak je transkripční faktor pozitivní a od daného místa transkripci spouští - opět je to snadná podmínka a zelený kus okraje může být místo na které nasedá ekvivalent RNA transkriptázy.

### 2.2.4. Speciální funkce

## 2.3. Buňky

# 3. Implementace

## 3.1. Použité technologie

# 4. Návod k použití

## 4.1. Kompilace

Na sestavení projektu je potřeba mít nainstalované následující programy a knihovny (knihovny se hledají pomocí `find_package` v CMake):

- kompilátor c++, který (dostatečně) podporuje c++23, například g++ 14 nebo novější nebo clang++ 18 nebo novější
- CMake 3.22 nebo novější
- sestavovací systém podporovaný cmake, například GNU make

## 4.1 Návod k použití — Kompilace

- knihovnu glm
- knihovny pro vulkan (včetně vulkan-hpp verze 1.4.309 nebo novější), typicky z LunarG Vulkan SDK (<https://vulkan.lunarg.com/>), ale pro linux bývají v repozitářích
- knihovnu vulkan memory allocator
- knihovnu boost 1.81 nebo novější (stačí komponenta math)

Vše lze dohromady systému Ubuntu nainstalovat následujícím bash skriptem, ale pravděpodobně je v repozitářích několik z těchto věcí v moc starých verzích:

```
sudo apt install build-essential cmake
sudo apt install vulkan-tools libvulkan-dev spirv-tools
sudo apt install libglfw3-dev libglm-dev libboost-dev

git clone https://github.com/GPUOpen-LibrariesAndSDKs/VulkanMemoryAllocator.git
cd VulkanMemoryAllocator
cmake -S. -Bbuild -DCMAKE_BUILD_TYPE=Release
cmake --build build
sudo cmake --build build -t install
```

Tento projekt pak jde naklonovat a zkompilovat následujícím bash skriptem:

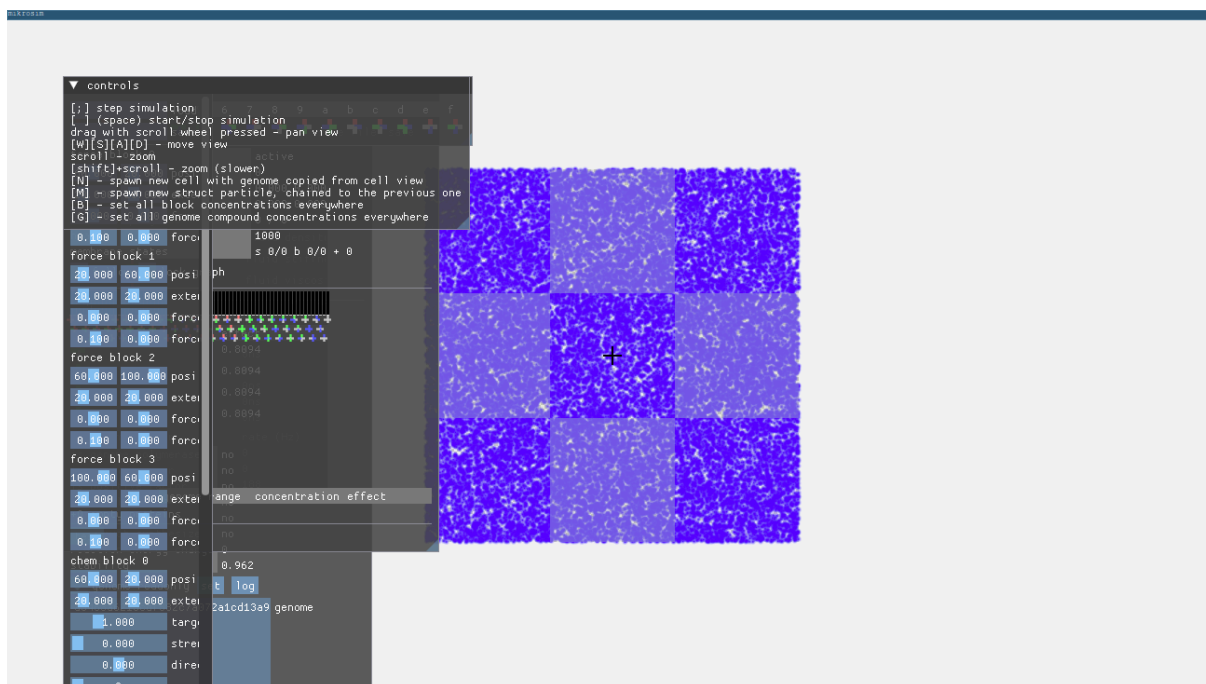
```
git clone --recurse-submodules https://github.com/nat-int/mikrosim.git
cd mikrosim
cmake -S. -Bbuild -DCMAKE_BUILD_TYPE=Release -DVULKAN_VALIDATION=OFF \
  -DCOMPILER_SHADERS=OFF
cmake --build build
```

Výsledný spustitelný soubor pak je `./out/mikrosim`, případně `./out/mikrosim.exe`, (relativně k cestě po příkazu `cd mikrosim`). Je ho potřeba spouštět ze složky `./out/`.

## 4.2. Návod k aplikaci

Po prvním spuštění vypadá aplikace přibližně takto:

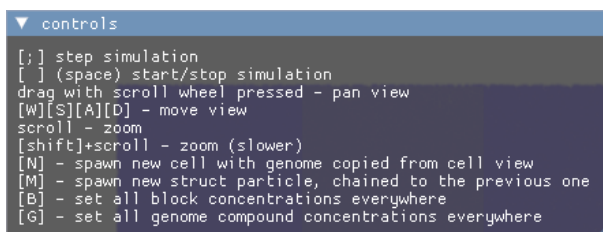
## 4.2 Návod k použití — Návod k aplikaci



Obrázek 4: aplikace po spuštění

Obsahuje několik podoken, se kterými se dá levým tlačítkem myši pohybovat a připínat k hlavnímu oknu nebo do sebe navzájem. Mimo podoken je vidět zobrazení simulace (modrý čtverec za okny).

### 4.2.1. okno „Controls“

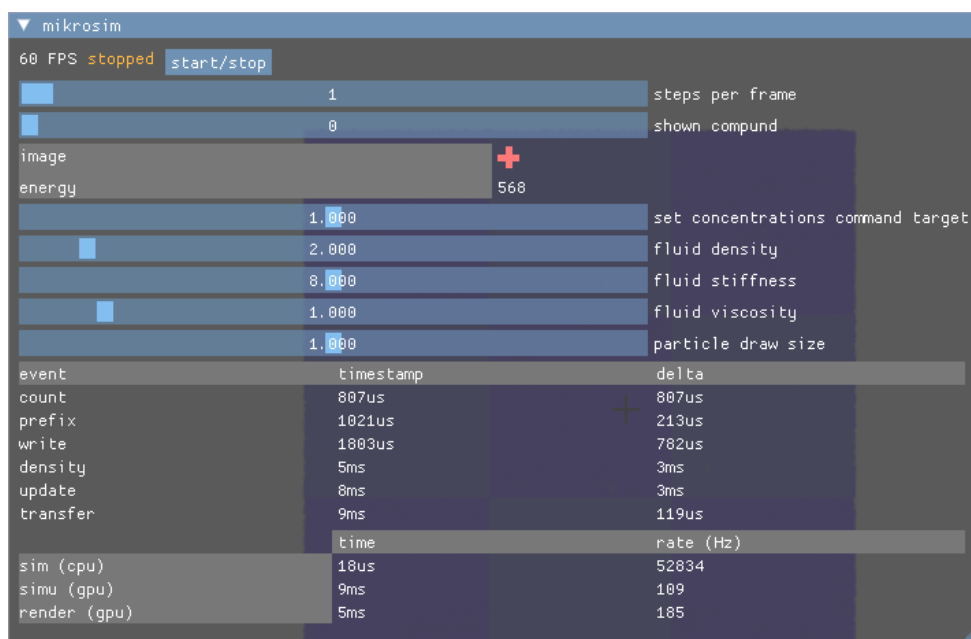


Okno „Controls“ obsahuje seznam kláves/akcí pro interakci se zobrazením simulace s popisem jejich efektu.



## 4.2 Návod k použití — Návod k aplikaci

### 4.2.2. okno „mikrosim“



Okno mikrosim začíná počtem snímků v poslední vteřině, poté text říkající, zda simulace běží a tlačítko, které simulaci spustí, nebo zastaví.

Na druhém řádku je posuvník, na kterém se nastavuje počet kroků simulace, které se provedou mezi každými dvěma snímky (když simulace běží).

Na třetím řádku je posuvník, na kterém se nastavuje látka, která se právě zobrazuje. Pod ním je obrázek dané látky a energie uložená v jejích vazbách.

Další posuvník nastavuje koncentraci, na kterou klávesy G a B nastavují (když se stiskne klávesa B když žádné okno nemá „focus“, tak se koncentrace všech proteinogéních látek nastaví na tuto hodnotu).

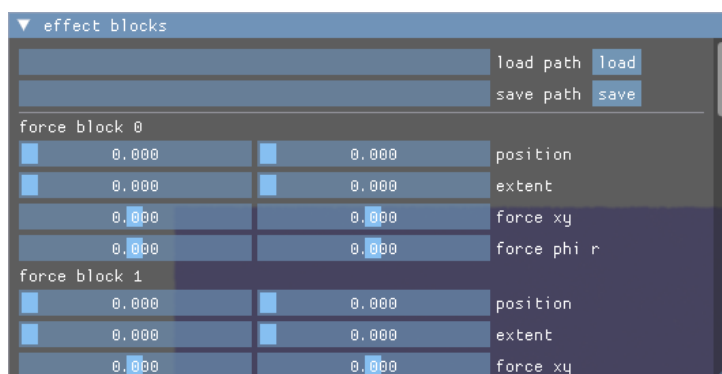
Následující 3 posuvníky nastavují vlastnosti tekutiny - hustotu, které se snaží dosáhnout, koeficient síly vyrovnávání hustoty a viskozitu.

Poslední posuvník nastavuje velikost částic při jejich vykreslování.

Pod posuvníky jsou tabulky s časy jednotlivých částí výpočtů simulace a vykreslování.

## 4.2 Návod k použití — Návod k aplikaci

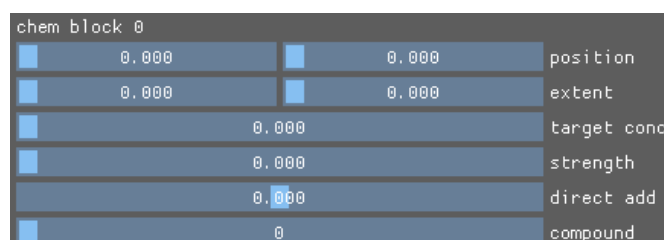
### 4.2.3. okno „effect blocks“



Toto okno obsahuje nastavení „efektových bloků“, kterými se dá ovlivňovat simulace. Existují jich 2 typy: silové („force blocks“) a chemické („chem blocks“). Silové bloky umožňují působit silou na částice v obdélníkové oblasti simulace. Chemické umožňují ovlivňovat koncentrace nějaké látky v obdélníkové oblasti simulace. Od obou typů jsou dostupné 4 bloky. Bloky se vykreslují přes zobrazení simulace s vysokou průhledností, silové bloky červeně a chemické zeleně.

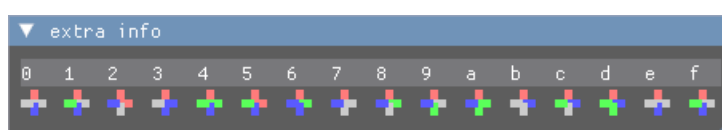
Na začátku jsou dvě textová pole s tlačítky - první umožňuje načíst nastavení všech bloků ze souboru, jehož cesta je napsána v textovém poli. Druhý umožňuje do souboru v textovém poli nastavení všech bloků uložit.

Pak následují nastavení silových bloků. Každé začíná pozicí a velikostí v osách x a y. Na třetím řádku posuvníků je nastavení homogení síly v osách x a y. Na čtvrtém řádku je nastavení síly kolem středu bloku (proti směru hodinových ručiček) a síla ke středu bloku. Síly jsou malé a tak chvilku trvá, než zapůsobí.



Poté následují nastavení chemických bloků. Stejně jako u silových bloků začínají dvěma řádky nastavení pozice a velikosti. Poté následuje cílová koncentrace, ke které se koncentrace přibližuje, poté síla, která určuje rychlost tohoto přibližování. Posuvník „direct add“ udává koncentraci, která se přičítá ve všech částicím v oblasti. Nakonec je posuvník určující látku, kterou blok zrovna ovlivňuje (čísla látek jsou stejná jako v okně „mikrosim“).

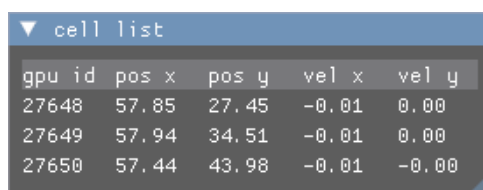
### 4.2.4. okno „extra info“



Okno extra info obsahuje tabulku pro převod mezi označením v genomu a látkou v proteinu.

## 4.2 Návod k použití — Návod k aplikaci

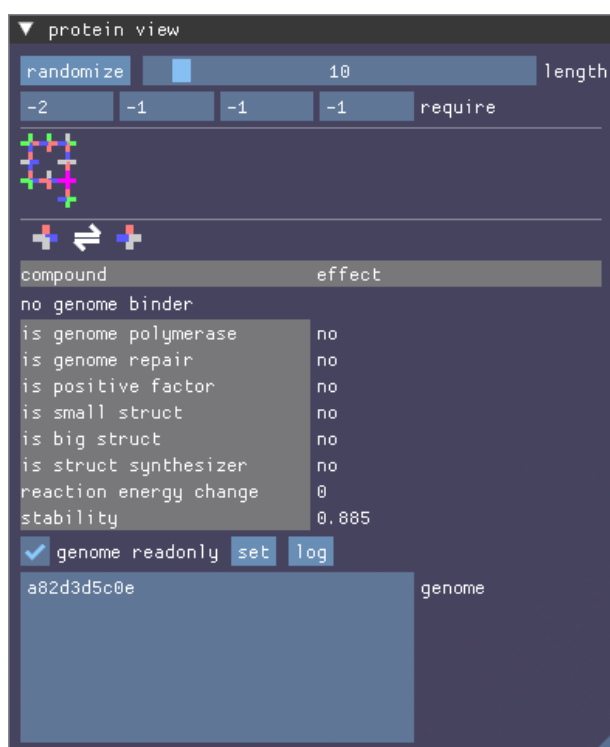
### 4.2.5. okno „cell list“



gpu id	pos x	pos y	vel x	vel y
27648	57.85	27.45	-0.01	0.00
27649	57.94	34.51	-0.01	0.00
27650	57.44	43.98	-0.01	-0.00


Okno cell list obsahuje seznam všech živých buněk s jejich identifikačním číslem, pozicí a rychlostí.


### 4.2.6. okno „protein view“



randomize ☐ 10 length

-2 -1 -1 -1 require





compound	effect
no genome binder	
is genome polymerase	no
is genome repair	no
is positive factor	no
is small struct	no
is big struct	no
is struct synthesizer	no
reaction energy change	0
stability	0.885

☒ genome readonly

a82d3d5c0e genome

### 4.2.7. okno „cell view“

### 4.2.8. zobrazení simulace

## 5. Závěr