TECHNICKÁ UNIVERZITA V KOŠICIACH
FAKULTA ELEKTROTECHNIKY A INFORMATIKY

Strelci na šachovnici

Inteligentné rozhodovacie systémy

2015/2016 Dominik Horňák, Richard Halčin, Matej Kvetko

Obsah

1. Úloha – Matej Kvetko

Našou úlohou bolo vložiť maximálny počet strelcov na šachovnicovú plochu o zadanej veľkosti **n**x**n** tak, aby sa navzájom neohrozovali. Podľa pravidiel šachu ohrozuje strelec všetky políčka na oboch diagonálach na ktorých stojí.

Pomocou výrokovej logiky sme vytvorili podmienky pre jednotlivých strelcov tak, aby mohli byť použité v SAT solveri. Pravidlá navyše spĺňajú pravidlá CNF.

2. Algoritmus – Matej Kvetko

$\bigvee_{i,j=1}^{i,j=n} p_{ij}$ **&**

definujeme všetky možnosti, nie je to nutná podmienka, je kvoli SAT solveru

(not(p_{ij}) v not(p_{km}))

po dvojiciach vynegujeme všetky ohrozenia strelcami pre každú možnú dvojicu strelcov

pre každé **(i,j != k,m) & (i+x=k) & (j+x=m) V (i+x=k) & (j-x=m) pre kazde x 0<(x+i)<=n & 0<(x+j)<=n**

Všetky podmienky sú v konjunkcii.

2.1. Príklad

11	12	13
21	22	23
31	32	33

(p11 v p12 v p13 v p21 v p22 v p23 v p31 v p32 v p33) &
(-p11 v –p22) & (-p11 v –p33) &
(-p12 v –p21) & (-p12 v –p23) &
(-p13 v –p22) & (-p13 v –p31) &
(-p21 v –p12) & (-p21 v –p32) &
(-p22 v –p11) & (-p22 v –p13) & (-p22 v –p31) & (-p22 v –p33) &
(-p23 v –p12) & (-p23 v –p32) &
(-p31 v –p22) & (-p31 v –p13) &
(-p32 v –p21) & (-p32 v –p23) &
(-p33 v –p22) & (-p33 v –p11)

3. Popis tried a funkcií v kóde programu – Dominik Horňák

Program je zložený z 3 častí, a to:

- Generátor pravidiel
- SAT – Solver, ktorý je kompletne pokrytý knižnicou minisat
- Grafické vyobrazenie, pre ktoré sme použili knižnicu wxWidgets.

3.1. Generátor pravidiel

Generátor je zložený z 3 tried Rule (pravidlo), Fields (polia), Generator.

3.1.1. Trieda Rule (pravidlo)

Táto trieda slúži na reprezentáciu pravidiel. Ukladá pravidlá vo forme A or B kde A a B sú literály vo formáte vhodnom pre knižnicu SAT-Solveru. Kôli minimalizácii kódu táto trieda môže reprezentovať pravidlá aj v špeciálnej forme std::vector<Minisat::Lit>, čo znamená že sa jedná o vektor už spomínaných literálov. Medzi jednotlivými literálmi sa nachádza operátor OR. Pre tvorbu objektu sa použije vhodný konštruktor, podľa toho či chceme vytvoriť základnú verziu pravidla alebo špeciálnu.

Zoznam funkcií s krátkym popisom:

- Rule(Minisat::Lit A, Minisat::Lit B) – konštruktor pre pravidlo v tvare A or B
- Rule(std::vector<Minisat::Lit>) – konštruktor pre pravidlo v špeciálnom tvare
- Minisat::Lit getA() – vráti kópiu literálu A
- Minisat::Lit getB() – vráti kópiu literálu B
- bool isSpecial() – ak je aktuálne pravidlo špeciálne vráti true, inak false
- std::vector<Minisat::Lit> getSpecial() – vráti kópiu vektora literálov, tak ako sú uložené v pamäti

3.1.2. Trieda Fields (polia)

Táto trieda reprezentuje polia na šachovnici. Samotné polia si generuje sama podľa rozmeru šachovnice zadaného v konštruktore. Taktiež táto trieda slúži ako akési spojenie medzi súradnicou polia a literálom ktorý reprezentuje stav tohto polia. Konkrétny algoritmus generácie nebude pokrytý v tejto dokumentácii, je však možné ho vydedukovať zo zdrojových súborov prístup ku ktorým je vo verejnom repozitári na portáli GitHub. Link k repozitáru bude umiestnený v závere tohto dokumentu.

Zoznam funkcií s krátkym popisom:

- Fields(int) – konštruktor, vytvorí celú šachovnicovú plochu vo formáte vhodnom pre účely nášho generátora s rozmerom strany šachovnice zadanom cez parameter
- Minisat::Lit getVar(int x, int y) – vráti literál umiestnený na pozícii X, Y
- int getX(Minisat::Lit) – vráti súradnicu X polia na ktorom sa nachádza literál vložený ako parameter
- int getY(Minisat::Lit) – vráti súradnicu Y polia na ktorom sa nachádza literál vložený ako parameter
- std::vector<Minisat::Lit> getVars() – vráti kópiu vektora všetkých literálov, pomocná funkcia určená pre generáciu špeciálneho pravidla
- int getNumVar() – funkcia ktorá vráti počet všetkých polí, no taktiež aj literálov

3.1.3. Trieda Generator

Táto trieda reprezentuje samotný generátor ktorý generuje pravidlá. Pravidlá sú generované hneď po vytvorení objektu pre šachovnicovú plochu s rozmerom zadaným do konštruktoru, takže jednotlivé pravidlá sú hneď po vytvorení objektu plne k dispozícii v dokonca až 2 formách.

Zoznam funkcií s krátkym popisom:

- Generator(int) – konštruktor, ktorý vytvorí objekt Fields s rozmerom zadaným ako parameter, ktorý je používaný ako pomocný objekt pri procese generovania pravidiel a následne vygeneruje všetky pravidlá používané pre riešenie nášho problému a ukladá ich v sebe vo forme vektora objektov typu Rule
- std::vector<std::string> getRulesForm1() – vráti vektor reťazcov s pravidlami vo formáte DIMACS, jednotlivé reťazce vo vektore reprezentujú jednotlivé riadky výsledného stringu ktorý je možné vložiť do ľubovoľného SAT-solvera na internete, podporujúceho tento formát
- std::vector<std::string> getRulesForm2() – vráti vektor reťazcov s pravidlami vo formáte akceptovanom na stránke http://logictools.org jedná sa o jednoduchý formát. Jednotlivé reťazce vo vektore reprezentujú jednotlivé riadky vo výslednom stringu ktorý je možné priamo vložiť do spomínaného online solvera
- int getDeskSize() – pre prípad potreby je tu funkcia ktorá vráti rozmer šachovnicovej plochy pre ktorú boli pravidlá generované
- Fields getVariables() – vráti kópiu objektu Fields, ktorý reprezentuje všetky polia spolu so všetkými literálmi na daných poliach. Potrebný pre vykresľovanie a použitie v Minisat Solveri

- std::vector<Rule> getRules() – vráti vektor všetkých vygenerovaných pravidiel vo forme objektov typu Rule, ktorý je designovaný tak aby bolo použitie pravidiel v Minisat Solveri čo najjednoduchšie

3.2. SAT Solver – Minisat

Táto knižnica je použitá tak vo forme ako bola prístupná na internete. Jedná sa o veľmi light SAT-Solver, ktorý povyhrával niekoľko súťaží v rýchlosti výpočtov. Jediné zmeny ktoré sme robili boli v rozbehaní knižnice pod windowsom, keďže knižnica bola písaná pre Linuxové distribúcie. Dokumentáciu k nej je možné nájsť na oficiálnej stránke projektu http://minisat.se

3.3. Grafické vyobrazenie

Pre GUI sme použili knižnicu wxWidgets. Prvky samotnej knižnice tu popisovať nebudem, spomeniem len spôsob ktorým boli vyťahované údaje z nášho generátora pre potreby vykresľovania a vypisovania dát.

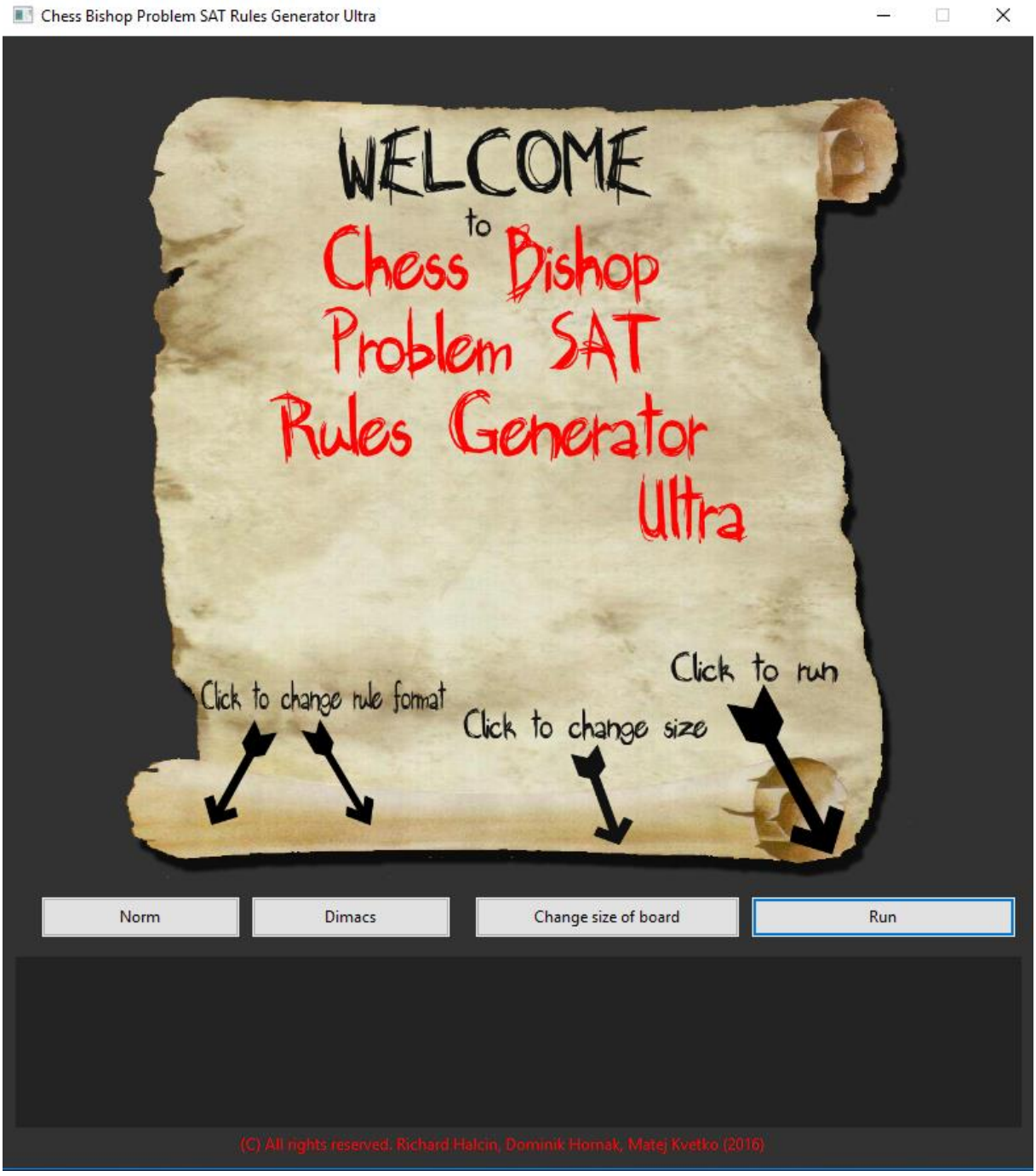
Použitie nášho generátora je veľmi jednoduché, stačí vytvoriť objekt typu Generator, kde ako vstupný parameter vložíme od používateľa načítaný požadovaný rozmer šachovnice. Následne si už vieme jednoducho vytiahnúť pravidlá v potrebnom formáte cez už popísané funkcie.

Pre Minisat Solver sa vytvára objekt typu Minisat::Solver do ktorého v ktorom sa vytvorí potrebný počet premenných (v závislosti od toho koľko premenných je v našom objekte Fields). Následne sa do neho vložia pravidlá vytiahnuté z vektora objektov typu Rule.
A nakoniec sa len spustí funkcia solve ktorá spustí algoritmus riešenia. Po dokončení riešenia je možné vytiahnúť všetky literály priamo zo solvera. Z literálov je možné vytiahnúť ich aktuálnu hodnotu true alebo false a použitím naších metód v triede Fields je možné jednoducho vytiahnúť súradnice jednotlivých polí pre ktoré je výsledok zo solveru true, čiže nachádza sa tam strelec.

4. Používateľská príručka – Richard Halčin

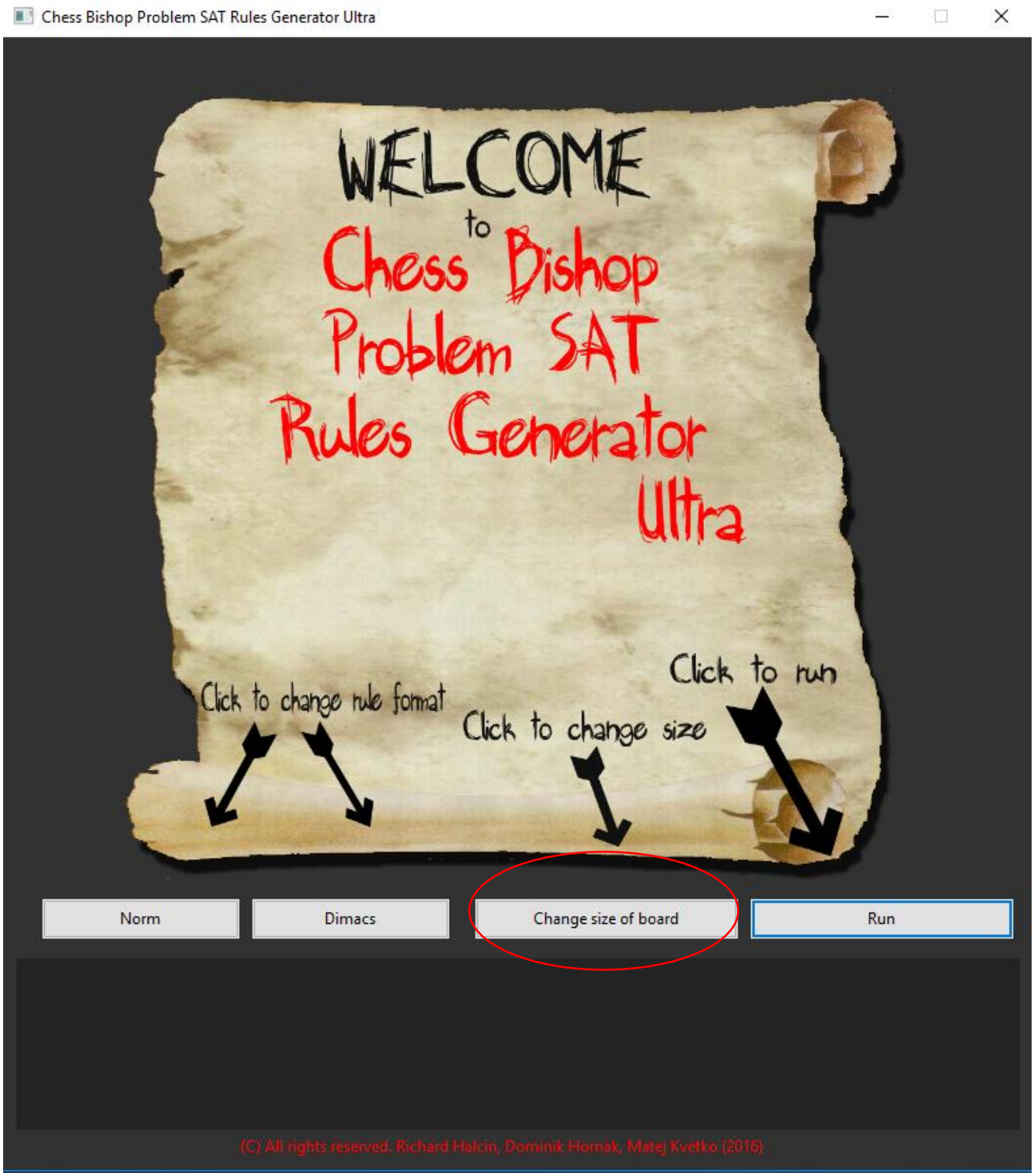
4.1. SPUSTENIE

1. Program spustime pomocou Chess-Bishop-Problem-SAT-Rules-Generator.exe

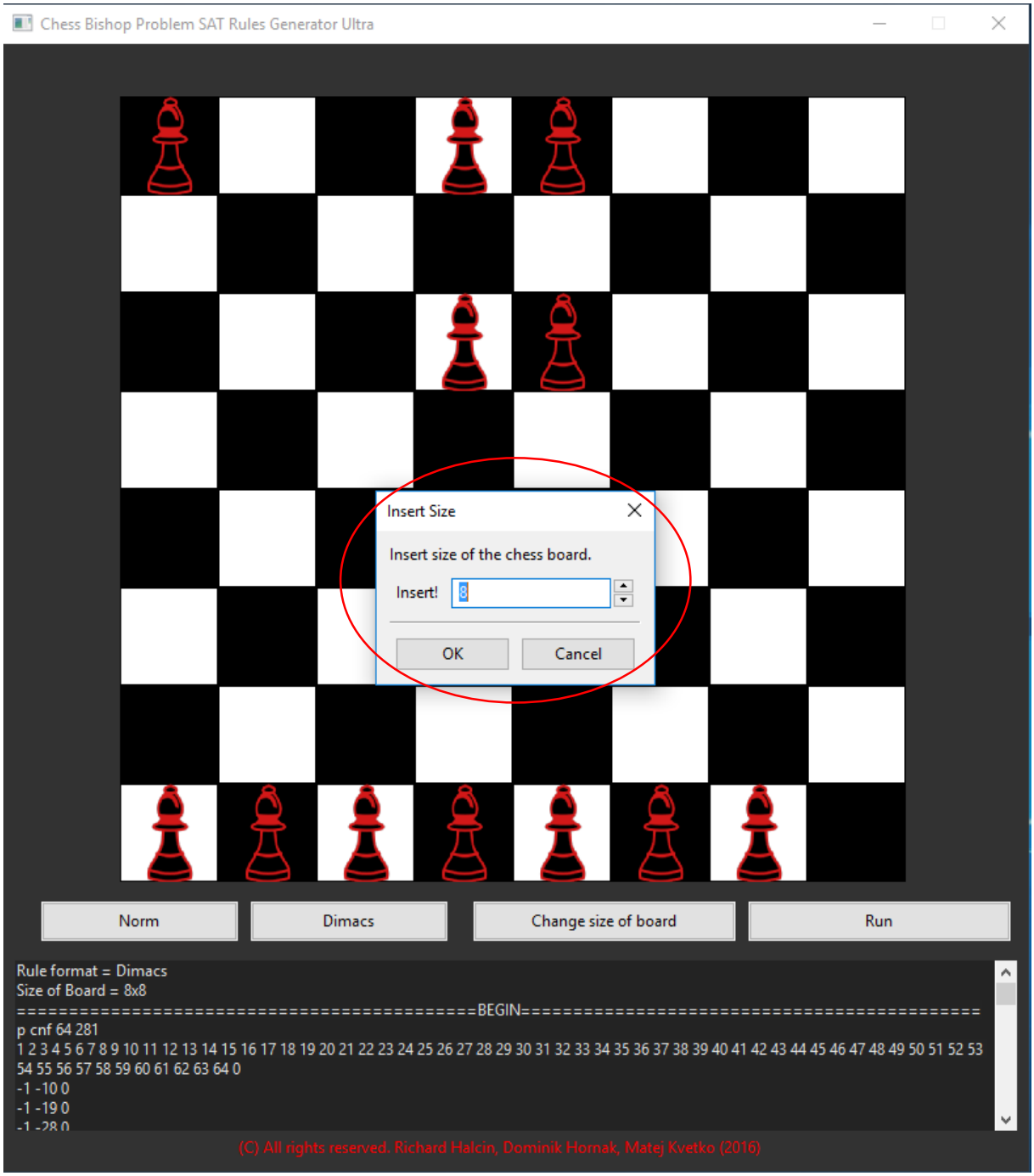


Toto je úvodná obrazovka ktorá sa zobrazí po spustení .exe súboru

4.2. ZMENA VELKOSTI POĽA



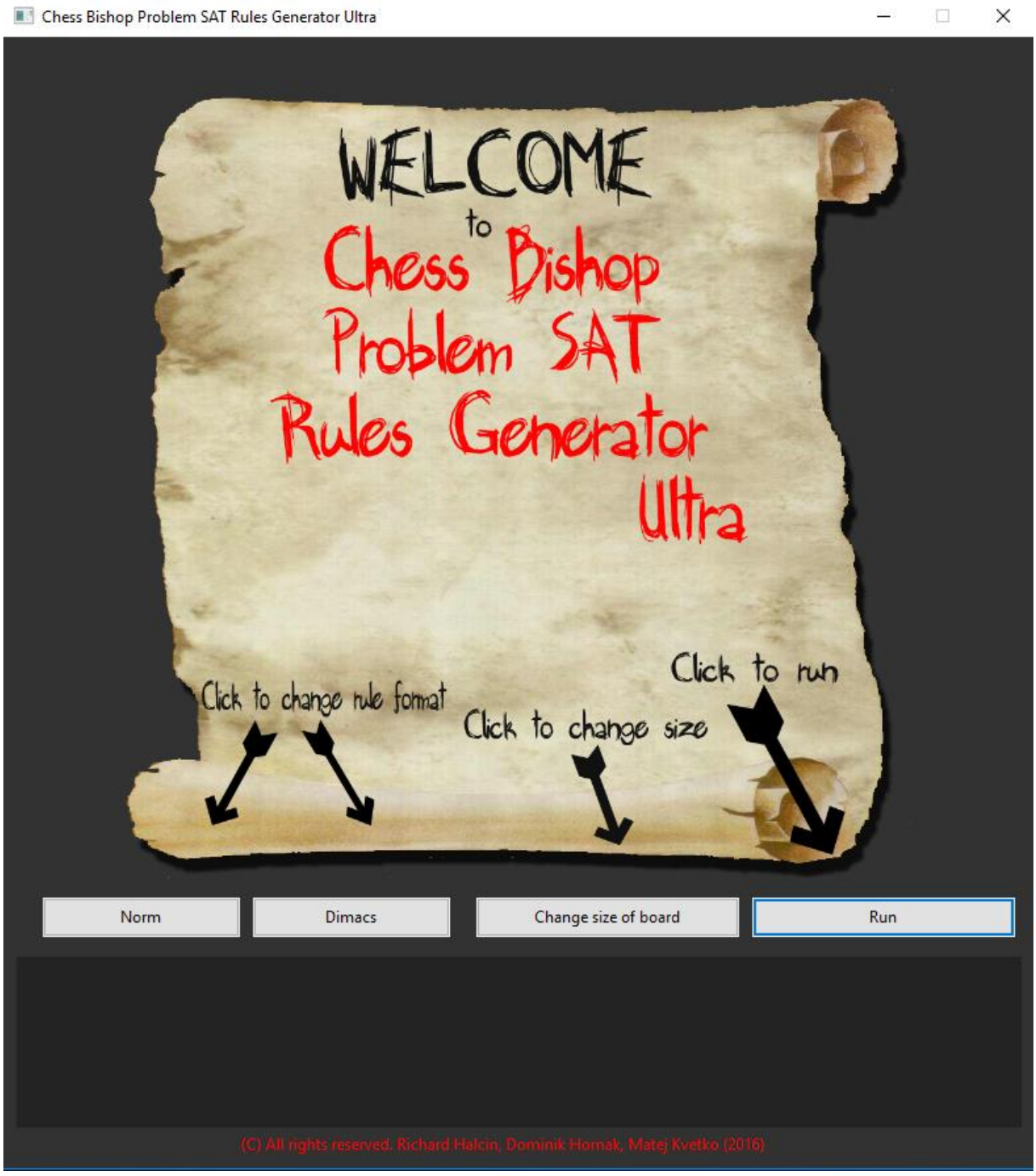
Po kliknutí na tlačidlo Change size of board sa užívateľovi otvorí nove okno v ktorom môže zadať veľkosť poľa.



Do tohto poľa môže užívateľ zadať ľubovoľne cele číslo od 1 až po max long.

Veľkosť poľa odporúčame zadávať s ohľadom na výkon vášho počítača.

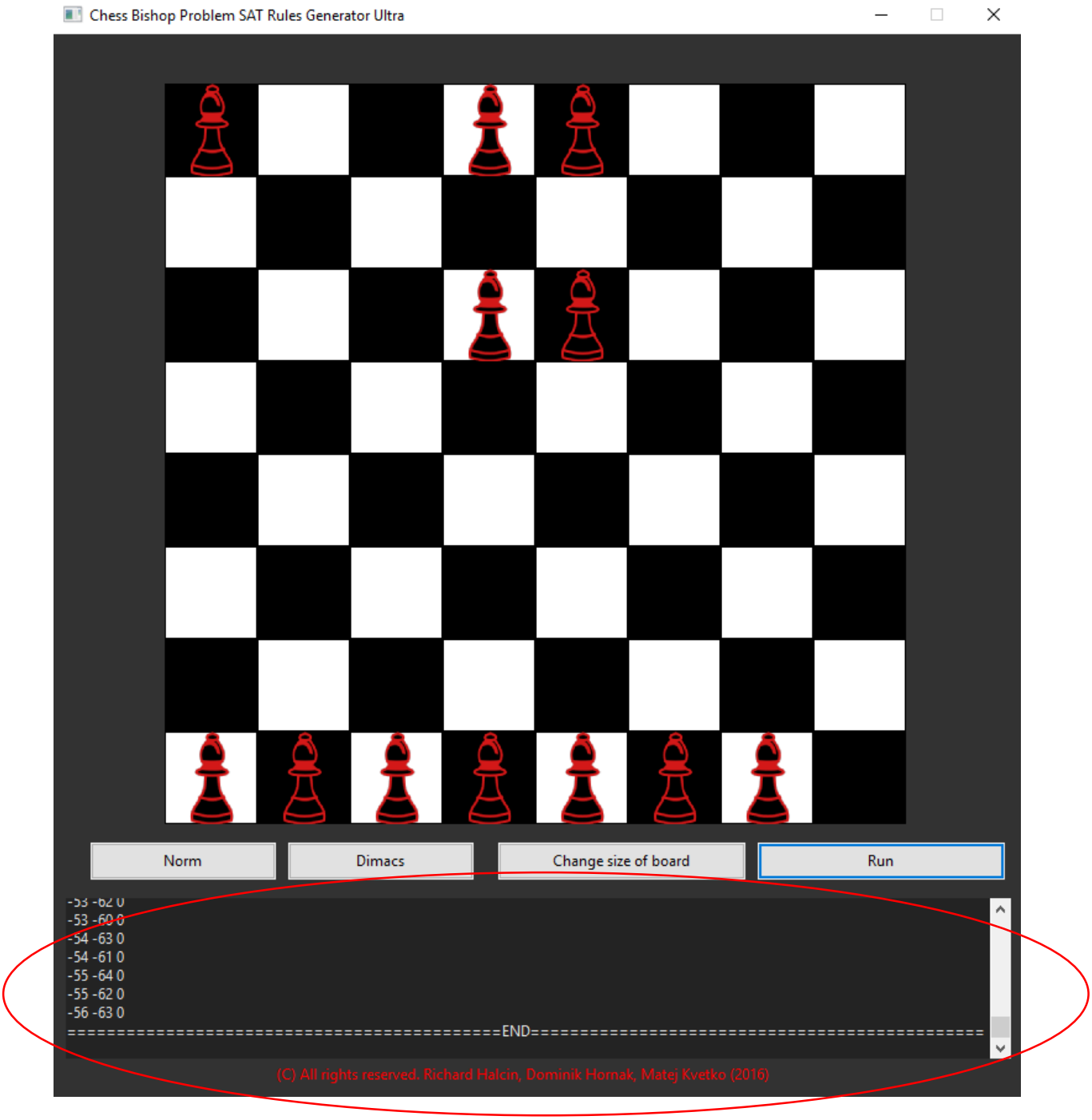
4.3. VYBER FORMÁTU PRAVIDLA



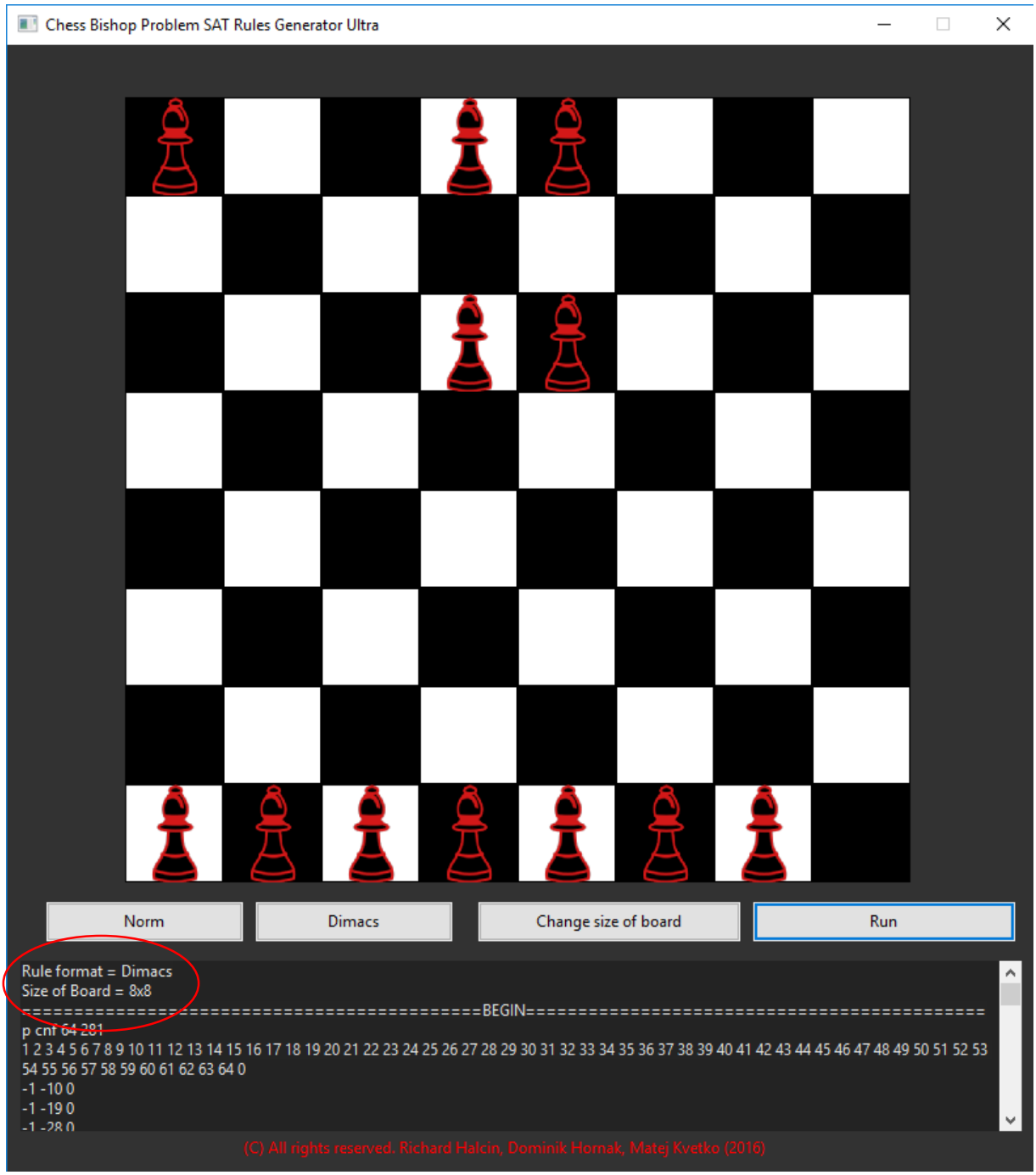
Užívateľ môže zmeniť formát pravidiel pomocou tlačidiel Norm a Dimacs.

4.4. SAMOTNE VYHODNOTENIE PRAVIDIEL

Po kliknutí na tlačilo Run program vyhodnotí pravidla vo vybranom formáte a užívateľom vybranej veľkosti poľa. Predefinovaná veľkosť je 1x1 a formát pravidla Dimacs. Následne program vykresli pole kde sú zobrazení strelci vo konkrétnych políčkach.



Ukážka pre veľkosť poľa 8x8 a formát pravidiel Dimacs. V spodnej časti môžete nájsť výpis pravidiel.



Pred samotným výpisom pravidiel program vypíše formát pravidiel a veľkosť poľa.

Link na GitHub:

https://github.com/Dominik-H/Chess-Bishop-Problem-SAT-Rules-Generator/