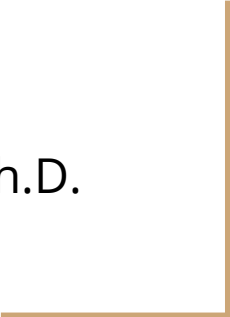


Aplikace pro demonstraci funkce Tabu prohledávání

Jaroslav Veselý

Vedoucí: doc. Ing. Petr Fišer, Ph.D.



Cíle práce

- Usnadnění studia/výuky předmětu MI-PAA
 - Pokročilé iterativní metody
- Vytvoření jednoduše spustitelné aplikace
 - Nástroj umožňující studentovi objevovat souvislosti a pochopit principy jednotlivých algoritmů
 - Nastavení parametrů algoritmu
 - Výběr problému a jeho instance (generátor, vlastní)
 - Grafické zpracování běhu algoritmu
 - Nezávislá implementace algoritmu a problému
 - Rozšiřitelnost

Postup

- Sběr a analýza požadavků na aplikaci
- Návrh UI
- Vytvoření prototypu
- Testování prototypu
- Implementace
 - algoritmy, problémy
- Uživatelské testování
- Úpravy vyplývající z testování

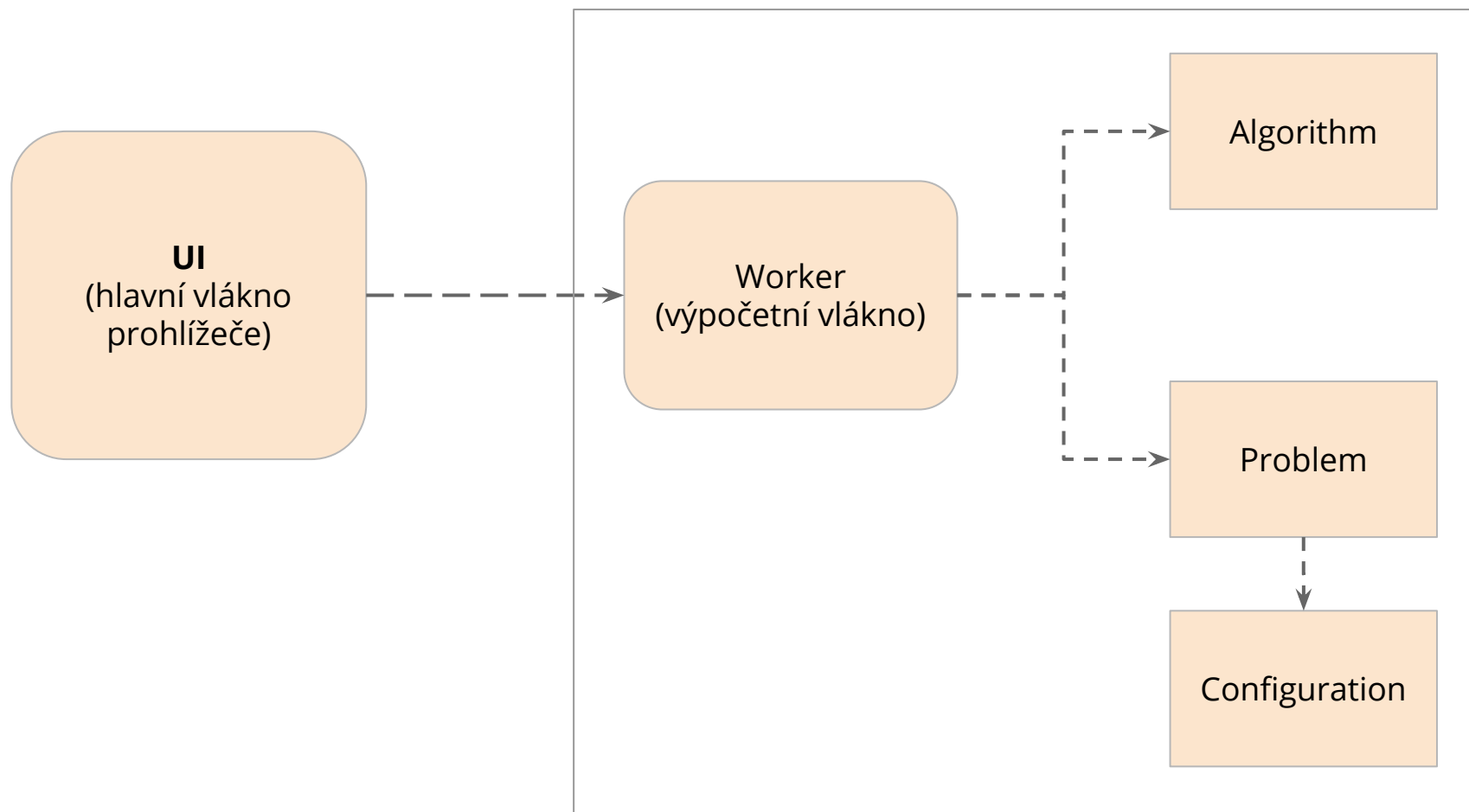
Kontext

- Aplikace je složena z výstupu tří DP
 - J. Veselý
 - Tabu prohledávání
 - Prototyp a společné UI, graf průběhu, SAT problém, sestavení aplikace
 - M. Kluzáček
 - Simulované ochlazování
 - Návrh UI - wireframe, SAT generátor, TSP, KNAP, MVC
 - A. Kugler
 - Genetický algoritmus
 - Testování prototypu, ETSP, modifikace grafu pro GA

Prostředí aplikace

- Webová aplikace (na základě průzkumu)
- Výpočty běží na straně klienta (JavaScript)
 - Web Workers
- Data se ukládají lokálně
 - IndexedDB, localStorage
- Framework Vue
- Bootstrap

Zprostředkování výpočtu



Tabu prohledávání

- Implementace nezávislá na konkrétním problému
- Nastavitelné parametry
 - Počet iterací
 - Část okolí pro výběr další konfigurace
 - Počet tabu stavů
 - Počet tabu změn v konfiguraci
 - Startovní stav (výchozí, náhodný)

Vykreslování grafu

- Knihovna *dc.js*
 - *inline svg + crossfilter*
 - efektivní zpracování změn dat
 - Vykreslování průběhu v reálném čase
 - Omezeno minimálním intervalem (graf a komunikace)
 - Porovnání běhů
 - Zvýraznění běhu při najetí na legendu
- Omezení webové aplikace
 - Vykreslování lze provádět pouze v hlavním vlákně

Advanced Iterative Meth: x

Secure | https://veselj43.github.io/dp-advanced-iterative-methods/#/

Simulated Annealing Genetic Algorithm Tabu Search

Iteration limit ?
300

Neighbor states to check ?
100 %

State tabu size ?
300

Changes tabu size ?
5

Starting configuration ?
Default

Status: Done ▶ Start run

Problem ?
Minimum vertex cover

Input instances ⬆ + 🗑

MVC_Example ℹ ⬇ 🗑

MVC80_250 ℹ ⬇ 🗑

Minimum vertex cover Minimize number of vertices

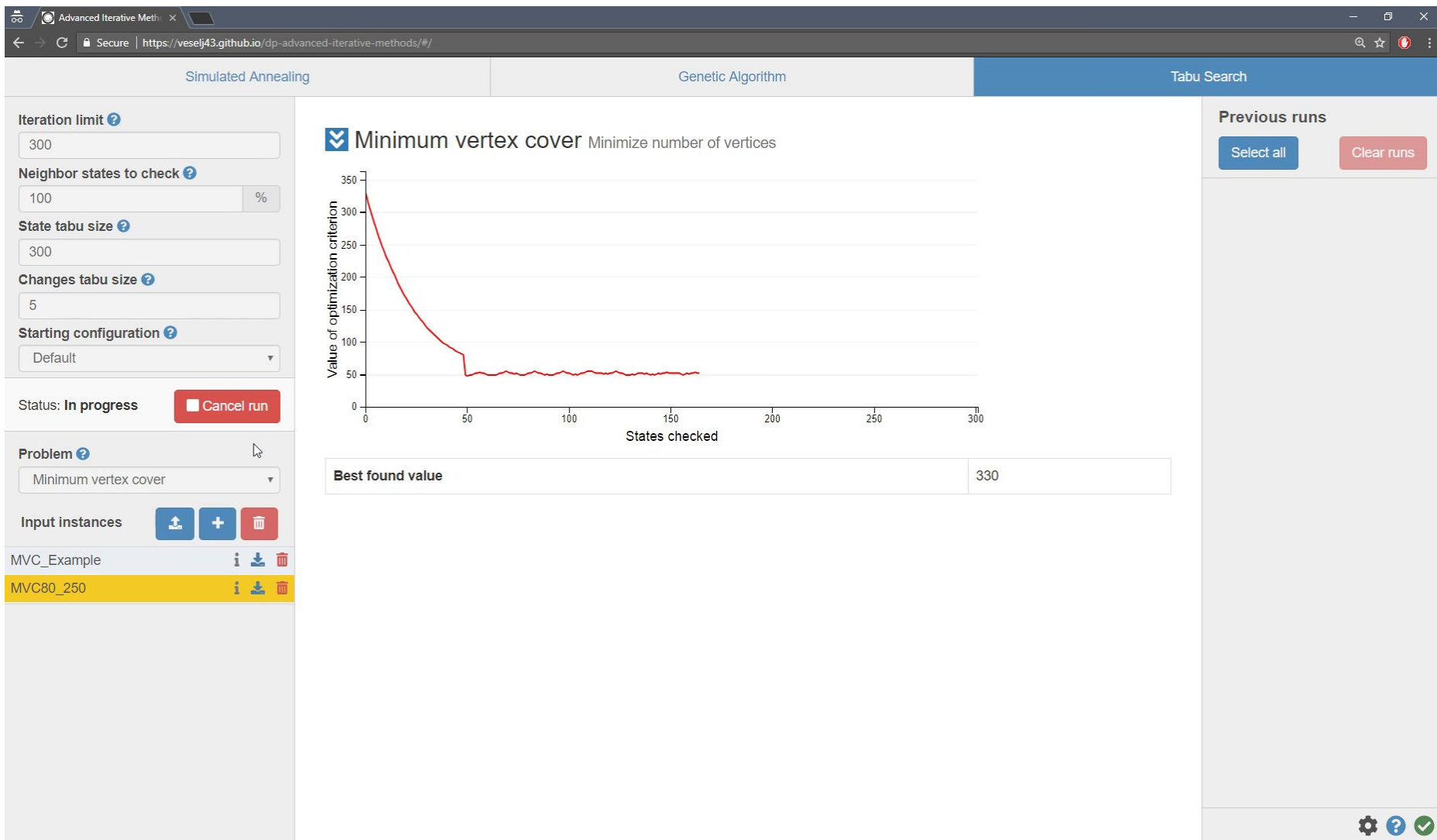
Start run or check previous runs to compare from the right panel.

Previous runs

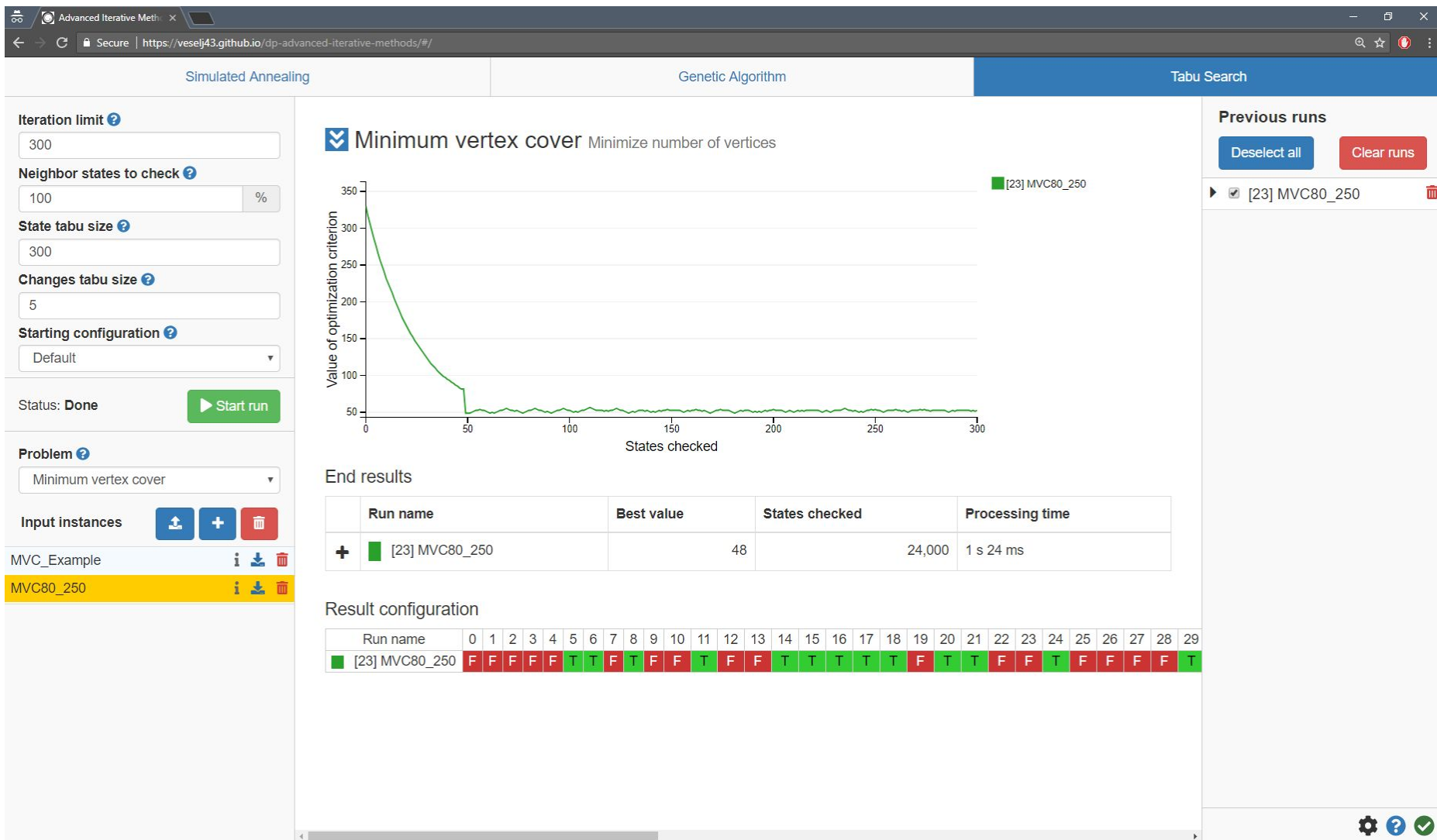
Select all Clear runs

⚙ ? ✓

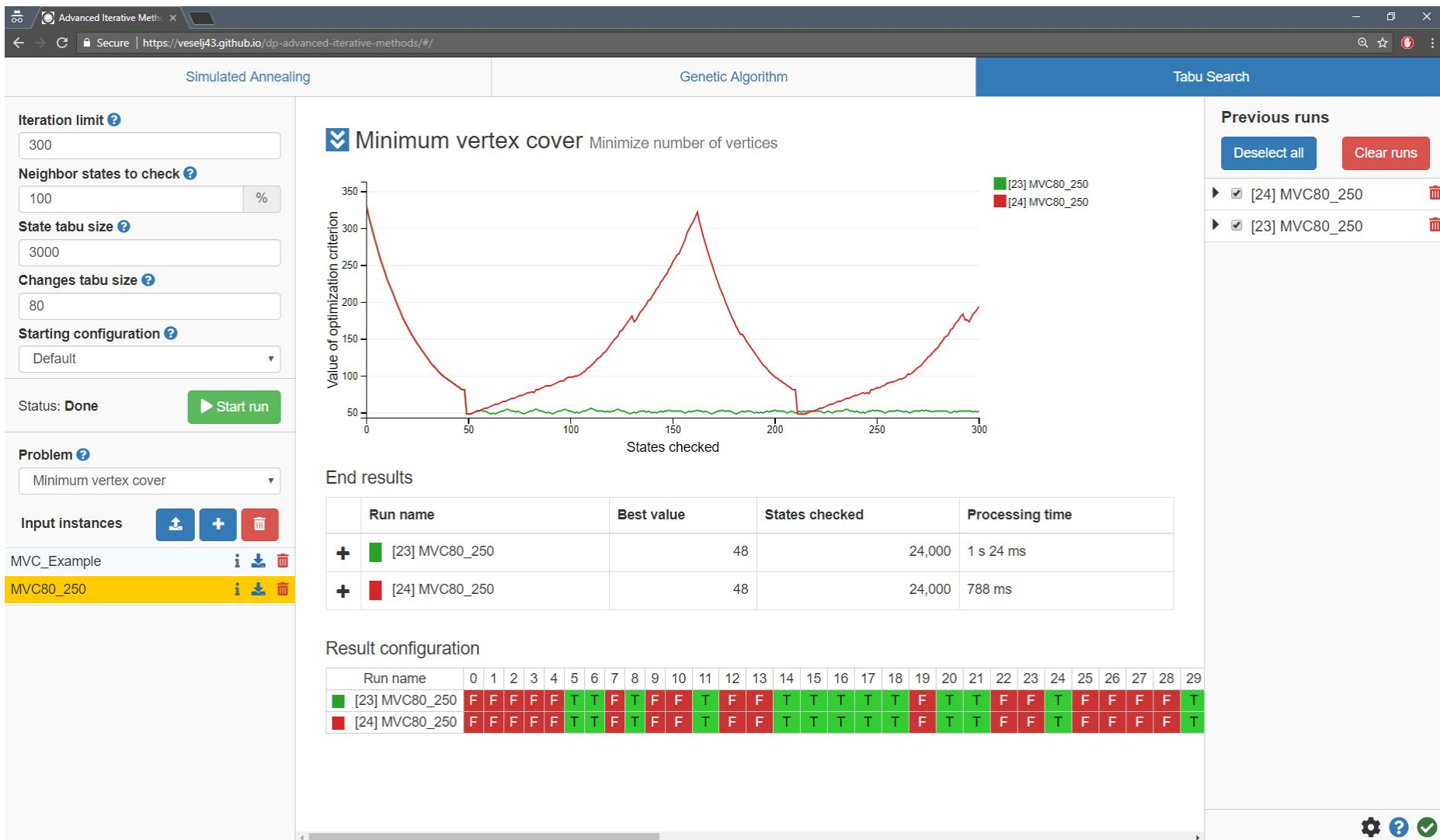
První načtení aplikace



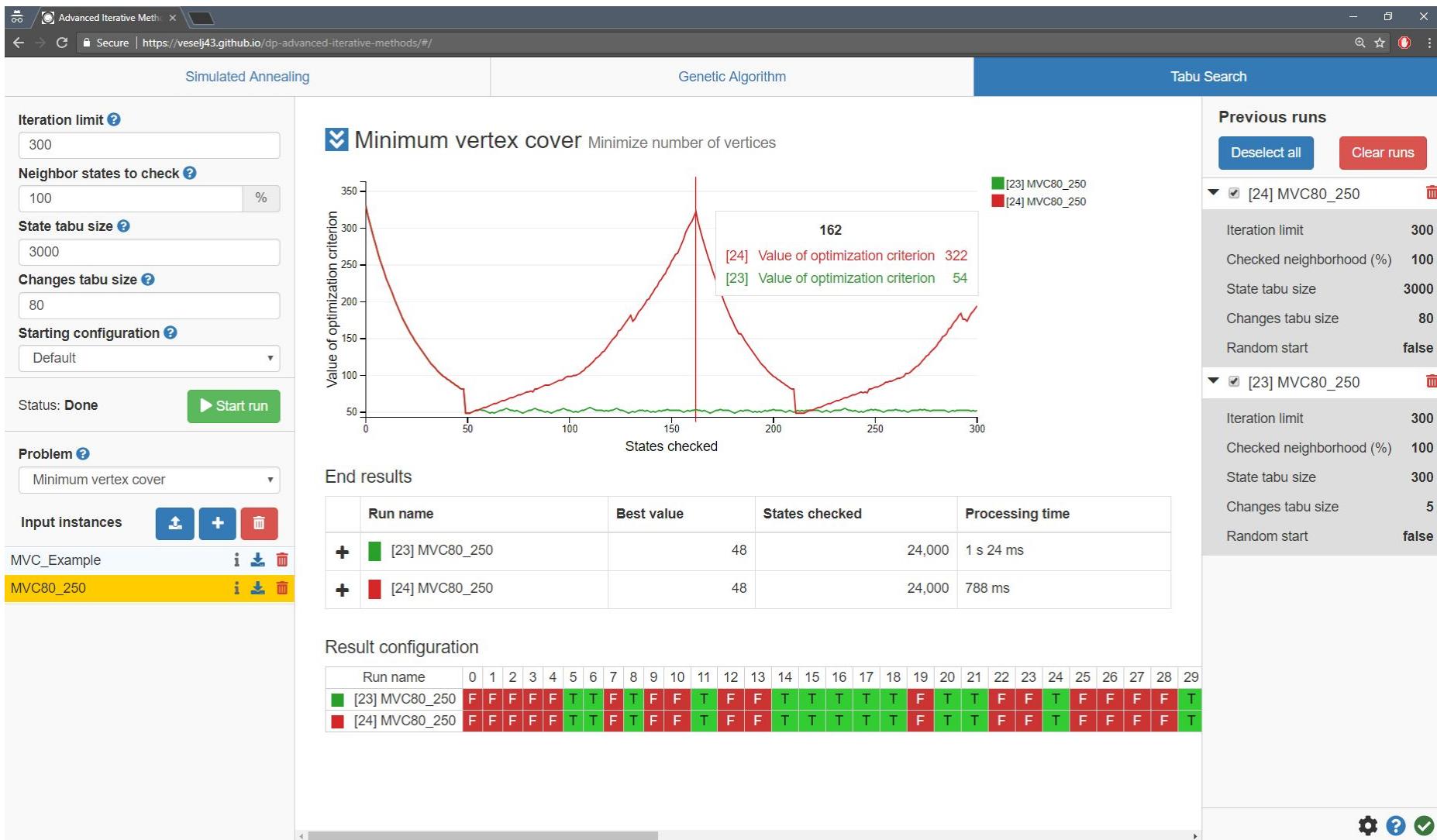
Průběh výpočtu



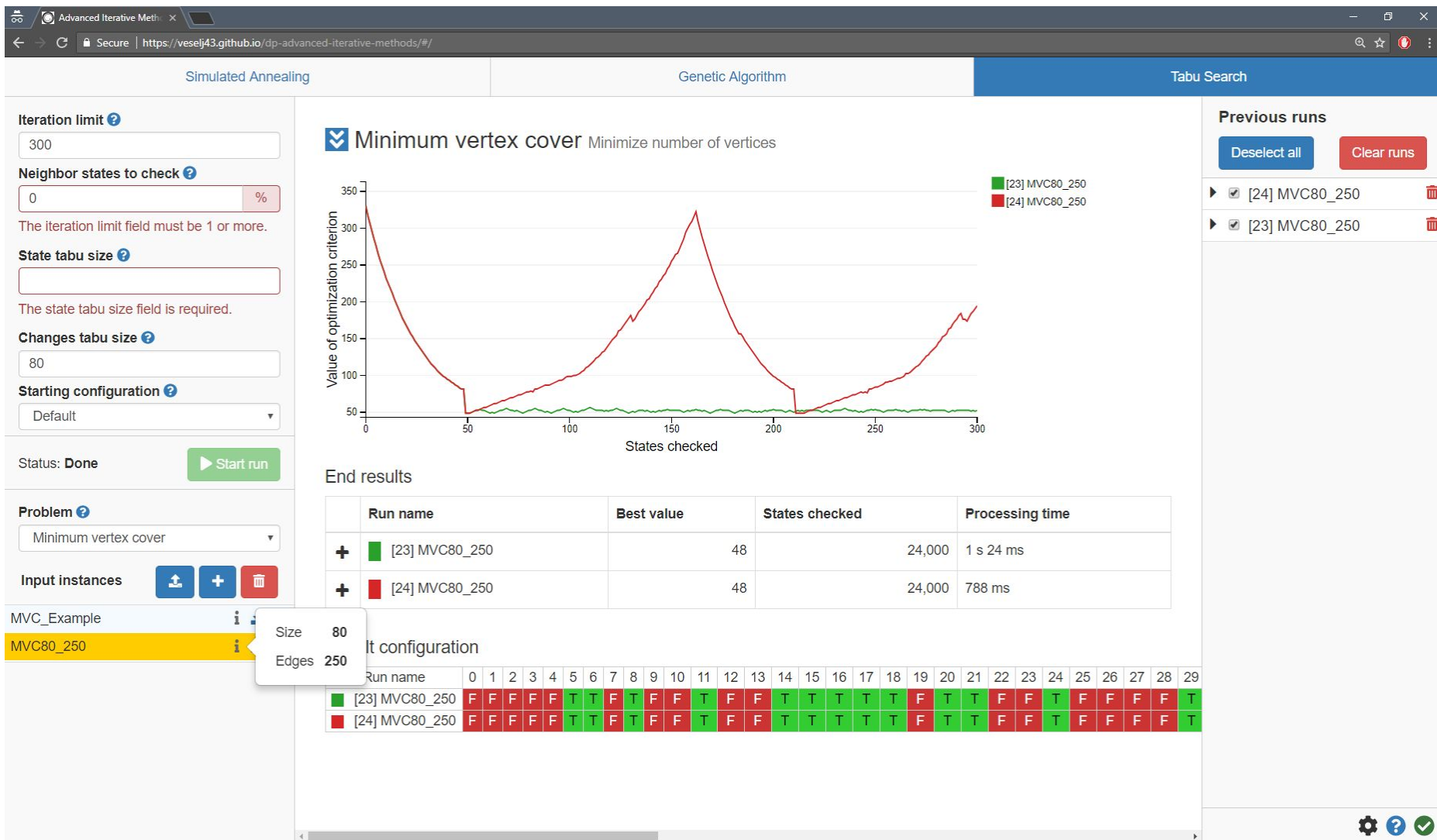
Zobrazení výsledku



Porovnání výsledků



Zobrazení hodnot a parametrů



Validace vstupu, zobrazení parametrů instance

Advanced Iterative Meth...

Secure | https://veselj43.github.io/dp-advanced-iterative-methods/#/

Simulated Annealing

Genetic Algorithm

Tabu Search

Iteration limit ?

300

Neighbor states to check ?

0

State tabu size ?

Changes tabu size ?

80

Starting configuration ?

Default

Status: Done

Start run

Problem ?

Minimum vertex cover

Input instances

MVC_Example

MVC80_250

How many neighbor configurations to check in one iteration

Minimum vertex cover

Minimize number of vertices

Value of optimization criterion

States checked

[23] MVC80_250

[24] MVC80_250

End results

	Run name	Best value	States checked	Processing time
+	[23] MVC80_250	48	24,000	1 s 24 ms
+	[24] MVC80_250	48	24,000	788 ms

Result configuration

Run name	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29
[23] MVC80_250	F	F	F	F	F	T	T	F	T	F	F	T	F	F	T	T	T	T	T	F	T	T	F	F	T	F	F	F	F	T
[24] MVC80_250	F	F	F	F	F	T	T	F	T	F	F	T	F	F	T	T	T	T	T	F	T	T	F	F	T	F	F	F	F	T

Previous runs

Deselect all

Clear runs

[24] MVC80_250

[23] MVC80_250

https://veselj43.github.io/dp-advanced-iterative-methods/#/help/Tabu#input

⚙️

?

✓

Tooltip

Advanced Iterative Meth: x

Secure | https://veselj43.github.io/dp-advanced-iterative-methods/#/help/Tabu

Back to application

Help Page

Simulated Annealing

Genetic Algorithm

Tabu Search

Problems

Knapsack

MAX-SAT

Travelling salesman

Euclidean travelling salesman

Minimal vertex cover

Tabu Search algorithm for this application

Concepts

Neighbor and neighborhood

Neighbor definition depends on problem configuration type.

There are two types of configurations in this application:

Bit array

- Represented as boolean array
- Implementation of `getNeighbour` method
 - Method parameter defines array `index` so that $0 \leq \text{index} < \text{array.length}$
 - Flip the bit on position `index` (`array[index] = !array[index]`)
 - Return the new configuration

Permutation

- Represented as permutation of element identifiers stored in array
- Implementation of `getNeighbour` method
 - Method parameter defines two array indexes so that $0 \leq \text{index1}, \text{index2} < \text{array.length}$
 - Flip the identifiers on defined array indexes (`swap(array[index1], array[index2])`)
 - Return the new configuration

Neighborhood is set of all neighbors from a given problem configuration.

Aspiration criteria

If cost of a tabu state is greater than the best cost yet found, consider the state as if it wasn't tabu.

Shrnutí obsahu práce

- Implementace Tabu prohledávání
- Implementace problému SAT
- *Vývoj UI*
 - Prototyp
 - Opravy na základě testování
 - Finální verze společné části a Tabu
- Výběr technologií a knihoven
- *Způsob a optimalizace vykreslování grafů*
- Návrh a optimalizace zprostředkování výpočtů
 - Vytvoření výpočetního vlákna + komunikace