Minimální deterministický konečný automat: 1. Determinizace

Termín odevzdání:	**13.12.2020 23:59:59**
Hodnocení:	**2.0000**
Max. hodnocení:	**2.0000** (bez bonusů)
Odevzdaná řešení:	1 / 50
Nápovědy:	0 / 2

Toto je první ze tří částí bodované programovací úlohy předmětu AAG.

Úkolem této bodované programovací úlohy je naimplementovat algoritmy, které vám umožní nalezení minimálního *deterministického konečného automatu*, který přijímá stejný jazyk jako zadaný libovolný *nedeterministický konečný automat s více počátečními stavy*. Cílem je tedy implementovat sadu tří funkcí v jazyce C++ pro determinizaci, odstranění zbytečných a nedosažitelných stavů a minimalizaci. Signatury těchto tří funkcí jsou:

- `DFA determinize ( const MISNFA & );`,
- `DFA trim ( const DFA & );` a
- `DFA minimize ( const DFA & );`.

V první části, tedy v této úloze, je úkolem naimplementovat pouze determinizaci libovolného nedeterministického automatu s více počátečními stavy. Druhá část (v samostatné úloze) naváže na první a bude testovat nejen determinizaci, ale i implementací funkce `trim`, která bude mít za úkol odstranit z deterministického automatu nedosažitelné a zbytečné stavy. Poslední část (taktéž v samostatné úloze) pak bude testovat nejen determinizaci a odstraňování stavů, ale také minimalizaci.

Vaším úkolem je tedy implementace třech funkcí. Budete je však implementovat postupně v následujících třech úlohách.

1. úloha testuje pouze funkci `determinize`. Ekvivalence automatů se testuje uvnitř testovacího prostředí. Na pojmenování stavů nezáleží. Stačí tedy, aby výsledkem byl ekvivalentní deterministický konečný automat. Funkce `trim` a `minimize` se nevolají. Můžete je tedy ponechat tak, jak jsou v ukázce.
2. úloha testuje funkce `determinize` a `trim`. Na zadaný automat na vstupu se zavolají obě funkce v tomto pořadí. Ekvivalence automatů po sekvenci volání `trim(determinize(input))` se testuje uvnitř testovacího prostředí. Na pojmenování stavů nezáleží. Stačí tedy, aby výsledkem byl ekvivalentní deterministický automat bez zbytečných a nedosažitelných stavů. Funkce `minimize` se netestuje. Můžete ji tedy ponechat tak, jak je v ukázce. **Algoritmus na odstranění zbytečných stavů podle přednášky je definovaný jen pro automaty přijímající neprázdný jazyk. V případě, že automat přijímá prázdný jazyk, definujte návratovou hodnotu algoritmu tak, že vrací ekvivalentní jednostavový automat bez přechodů.**
3. úloha testuje všechny tři funkce. Na zadaný automat na vstupu se použije operace `minimize(trim(determinize(input)))` a je potřeba, aby výsledkem byl ekvivalentní **minimální** deterministický automat. Na pojmenování stavů nezáleží.

Úloha je nastavena bodově tak, že za správnou implementaci determinizace dostanete 2 body. Za přidání implementace odstranění zbytečných a nedosažitelných stavů (v další úloze) další jeden bod. Pokud přidáte i minimalizaci (v další úloze), můžete dostat další 2 body.

Vstupem, resp. výstupem, algoritmů jsou konečné automaty v podobě struktur `MISNFA`, resp. `DFA` reprezentující nedeterministický konečný automat s více počátečními stavy, resp. deterministický konečný automat. Tyto struktury jsou definovány v testovacím prostředí (ve svém úkolu je tedy nedefinujte), viz ukázka kódu v přiloženém archivu. Pro zjednodušení jsou stavy definovány jako hodnoty typu `int` a symboly abecedy jako hodnoty typu `char`.

Porovnání automatů s referenčním výsledkem se provádí přes testování izomorfismu přechodových funkcí minimálních deterministických konečných automatů. Váš výstup se tedy může of referenčního lišit maximálně v pojmenování stavů, jinak bude vyhodnocen jako nesprávný.

Je zaručeno, že na vstupu funkce `determinize` budou validní nedeterministické konečné automaty s více počátečními stavy. Budou splňovat následující vlastnosti:

- Množiny stavů (`MISNFA::m_States`), počátečních stavů (`MISNFA::m_InitialStates`) a symbolů abecedy (`MISNFA::m_Alphabet`) budou neprázdné.
- Počáteční a koncové stavy z množin `MISNFA::m_InitialStates` a `MISNFA::m_FinalStates` budou také prvky množiny stavů `MISNFA::m_States`.

- Pokud nebude pro nějaký stav `q` a symbol abecedy `a` definovaný přechod v automatu, pak v `MISNFA::m_Transitions` nebude klíč `(q, a)` vůbec existovat.
- V přechodové tabulce `MISNFA::m_Transitions` se vyskytují také jen takové symboly a stavy, které jsou specifikovány v množině symbolů abecedy a v množině stavů.

Výsledný `DFA` musí také splňovat podmínky definice automatu, tedy musí platit to samé co výše pro MISNFA (až na zřejmé změny kvůli rozdílným definicím počátečního stavu a přechodové funkce). Na vstupu funkcí `trim` a `minimize` budou také validní automaty, splňující požadavky na to, aby to byl deterministický konečný automat.

Odevzdávejte zdrojový soubor, který obsahuje implementaci požadovaných funkcí pro danou úlohu. Do zdrojového souboru přidejte i další Vaše podpůrné funkce či datové struktury. Funkce budou volané z testovacího prostředí, je proto důležité přesně dodržet zadané rozhraní funkce. Nedodržení povede k chybě při kompilaci. Za základ pro implementaci použijte kód z ukázky níže. V kódu chybí vyplnit implementace tři zmíněných funkcí (a případné další podpůrné funkce či datové struktury). Ukázka obsahuje testovací funkci `main`, uvedené hodnoty jsou použité při základním testu. Všimněte si, že vkládání hlavičkových souborů, struktur `DFA` a `MISNFA` a funkce `main` jsou zabalené v bloku podmíněného překladu (`#ifdef/#endif`). Ponechte definice těchto struktur, include direktivy preprocesoru i funkci `main` v blocích podmíněného překladu i v odevzdávaném zdrojovém souboru, jinak Váš program nepůjde zkompilovat.

Podmíněný překlad Vám zjednoduší práci. Při kompilaci na Vašem počítači můžete program normálně spouštět a testovat. Při kompilaci na Progtestu však vše uvnitř těchto bloků "zmizí", tedy nebude kolidovat s hlavičkovými soubory, již definovanými strukturami či funkcí `main` testovacího prostředí.

Pro základ implementace můžete využít soubor ke stažení níže v sekci Vzorová data. Tento soubor obsahuje také několik základních testů, mějte však na paměti, že výsledky Vaší implementace se mohou lišit v pojmenování stavů. Testy jsou nastaveny podle výsledků, které dává jedno z referenčních řešení. Možná si je tedy budete muset upravit.

Váš program bude spouštěn v omezeném testovacím prostředí. Je omezen dobou běhu (limit je vidět v logu referenčního řešení) a dále je omezena i velikost dostupné paměti.

Poznámky:

- Potřebné algoritmy pro tento úkol byly probrány v přednáškách a na cvičení.
- Aby byl automat minimální, musí splňovat určité vlastnosti.
- Mějte na paměti, že vstupy jsou různé validní automaty podle definice z přednášky. Koncový stav může být například jen jeden a to v nedosažitelné části automatu (tedy jazyk automatu může být prázdný) nebo také nemusí být koncový žádný.
- Vstupem funkce `determinize` je vždy struktura `MISNFA`, tedy formálně nedeterministický konečný automat s více počátečními stavy.
- Vstupem všech funkcí je vždy validní konečný automat.
- Pořadí testů je zvoleno schválně. Protože první test testuje pouze determinizaci, stačí implementovat pouze příslušnou funkci. V další fázi testů pak stačí přidat implementaci odstranění nedosažitelných a zbytečných stavů. Poslední fáze testů pak testuje i minimalizaci.
- Ukázkové vstupy po determinizaci vrací úplný DKA. Pokud vrátíte neúplný, podle algoritmu z přednášky, váš automat bude přijat také.
- Časový limit je nastaven poměrně benevolentně. Měl by postačit i pro naivní implementaci algoritmů podle přednášky.
- Dokumentace formátu zápisu KA z nápověd je **zde**.

Vzorová data:	**Download**

☐ **Referenční řešení**

1	**08.12.2020 20:21:47**	**Download**
Stav odevzdání:	Ohodnoceno	
Hodnocení:	2.0000	

- **Hodnotitel: automat**
  - Program zkompilován
  - Test 'Zakladni test (determinizace)': Úspěch
    - Dosaženo: 100.00 %, požadováno: 100.00 %
    - Celková doba běhu: 0.004 s (limit: 15.000 s)
    - Úspěch v závazném testu, hodnocení: 100.00 %
  - Test 'Test meznich hodnot (determinizace)': Úspěch
    - Dosaženo: 100.00 %, požadováno: 100.00 %
    - Celková doba běhu: 0.001 s (limit: 14.996 s)
    - Úspěch v závazném testu, hodnocení: 100.00 %
  - Test 'Test nahodnymi daty (determinizace)': Úspěch
    - Dosaženo: 100.00 %, požadováno: 50.00 %

        - Celková doba běhu: 0.968 s (limit: 14.995 s)
        - Úspěch v závazném testu, hodnocení: 100.00 %
    - Celkové hodnocení: 100.00 % (= 1.00 * 1.00 * 1.00)
- Celkové procentní hodnocení: 100.00 %
- Celkem bodů: 1.00 * 2.00 = 2.00

		Celkem	Průměr	Maximum	Jméno funkce
SW metriky:	Funkce:	**9**	--	--	--
	Řádek kódu:	**124**	**13.78 ± 13.88**	**46**	`main`
	Cyklomatická složitost:	**18**	**2.00 ± 1.05**	**4**	`determinize`