Cpt S 516 Homework # 4 Solutions

1. Let $P(n_a, n_b, n_c)$ be a Presburger formula defined as below:

$$\exists k_1 \geq 0 \exists k_2 \geq 0. n_a = k_1 + k_2 \wedge n_b = k_1 + k_2 \wedge n_c = k_1.$$

Then, $\#(L) = \{(n_a, n_b, n_c) : P(n_a, n_b, n_c)\}$.

2. Build a PDA M' as follows. M' faithfully simulates M. Additionally, whenever M pushes (resp. pops) a, M' reads an input symbol p_a (resp. q_a). Whenever M pushes (resp. pops) b, M' reads an input symbol p_b (resp. q_b). M' accepts if it is at the end of the input. We use L' to denote the context-free language accepted by M'. L' is also a semilinear language. Define $L = \{w : -2\#_{p_a} + 3\#_{p_b} < 0 \vee 3(-2\#_{p_a} + 3\#_{p_b}) < (\#_{p_a} - \#_{q_a}) + (\#_{p_b} - \#_{q_b})\}$. (Notice that, in the problem statement, money x is $-2\#_{p_a} + 3\#_{p_b}$, n is $\#_{p_a} - \#_{q_a}$, m is $\#_{p_b} - \#_{q_b}$.) Notice that L is a commutative semilinear language. So, $L \cap L'$ is also a semilinear language. Notice that M is not stable iff $L \cap L' \neq \emptyset$. The result follows from the decidability of testing emptiness of a semilinear set $L \cap L'$.

3. Define $P(s, x_a, x_b, x_c, s', x'_a, x'_b, x'_c)$ to be: from state s to s', M performed $x'_a - x_a$ number of activities a, $x'_b - x_b$ number of activities b, $x'_c - x_c$ number of activities c. Since $L_{ss'}$, the language on $\{a, b, c\}$ accepted by M when moving from s to s', is a regular language, clearly P is Presburger (there are only finitely many s and s') and transitive. Define P' as $x_a = x_b = x_c \wedge P(s, x_a, x_b, x_c, s', x'_a, x'_b, x'_c) \wedge x'_a = x'_b = x'_c$. P' is also Presburger and transitive. You need only check whether there is a chain of P' from the initial state.

4. Let p be an execution sequence of M. We use n_a, n_b, n_c to denote the number of a's, b's, and c's on p. The set of all such p's form a regular language (hence semilinear), so we have a Presburger formula $P(n_a, n_b, n_c)$ to exactly characterize the relationship on n_a, n_b, n_c. Notice that the constraint on t_a, t_b, t_c are exactly $n_a < t_a < 2n_a \wedge 2n_b < t_b < 3n_b \wedge 3n_c < t_c < 4n_c$. We use $P'(t_a, t_b, t_c)$ to denote the property that the absolute values between t_a, t_b, t_c on the sequence is not more than 100. It can be shown that M is time-fair iff the following formula is satisfiable:

$P(n_a, n_b, n_c) \wedge n_a < t_a < 2n_a \wedge 2n_b < t_b < 3n_b \wedge 3n_c < t_c < 4n_c \wedge P'(t_a, t_b, t_c)$.

The decidability follows from the decidability of mixed linear formulas (see Notes).

5. We use L_1 to denote the set of all words α such that α is a prefix of some word accepted by M. Clearly, L_1 is context-free and hence is semilinear. Define L_2 to be all words α such that α satisfies $\neg P$. Clearly, L_2 is commutative and semilinear. The result follows since the question-part is equivalent to the emptiuness of $L_1 \cap L_2$, which is still a semilinear language.

6. By assumption, we assume that algorithm REACH solves the following *point-to-point* reachability problem for VASS:

Given: a VASS and two configurations (p, v) and (q, u),

Question: can configuration (p, v) reach configuration (q, u) ?

Now, we construct an algorithm PRES-REACH to solve the following problem (called Presburger-reachability):

Given: a VASS G, two states p and q, and two Presburger formulas P and Q, Question: are there vectors v and u (in $\mathbf{N}^n$ of course) such that $P(v)$ and $Q(u)$ are true, and configuration (p, v) reaches configuration (q, u) ?

PRES-REACH works as follows. Given the above instance of the Presburger-reachability problem, we first construct a new VASS G'. Without loss of generality, assume that the given Presburger formulas P and Q define two linear sets (also denoted by P and Q), respectively. That is, $P = \{u_0 + t_1u_1 + \cdots + t_ku_k : t_i \geq 0\}$ and $Q = \{v_0 + t_1v_1 + \cdots + t_kv_k : t_i \geq 0\}$, for some constant vectors u_i and v_i (and some k). The VASS G' works as follows. It has an initial state s_0 and counters $(x_1, \cdots, x_n)$. By using additional states, G' computes as follows: it first increments the counters by (vector) u_0 and then, for each i, increments the counters by u_i for 0 or more times (nondeterministic chosen). Then, it starts to simulate G from state p. When G enters state q (not neccessarily the first time), G' nondeterministically forces G to suspend and then starts the following session: By using additional states, decrements the counters by (vector) v_0 and then, for each i, decrements the counters by v_i for 0 or more times (nondeterministic chosen). Then, G' enters a final state s_f. Now, PRES-REACH calls algorithm REACH on VASS G' and see whether $(s_0, \mathbf{0})$ can reach $(s_f, \mathbf{0})$ in G'. Then, PRES-REACH returns the result (yes/no) accordingly.

7. One can use asynchronous counter increment and counter tests in ADTA to simulate a two-counter machine (with counter A and B):

$A++$ by X_A++;
$A--$ by Y_A++;
$B++$ by X_B++;
$B--$ by Y_B++;
and tests of counters A and B can be simulated by counter tests (e.g., compare X_A and Y_A) in ADTA.
8. Many ways to solve the problem. I believe in the original paper of Alur and Dill, there was a construction. Also, for parametric clock constraints, I have a proof (sorry, I couldn't find in which paper).