Infinite Automata 2025/26

Lecture Notes 9

Henry Sinclair-Banks

Lemma 8.8. Reachability in safe counter-stack automata is PSPACE-hard.

Proof sketch. Recall, from Lecture 8, that we were using the following two-round instance of the subset-sum game decision problem to outline the main ideas behind the proof of Lemma 8.8.

$$(\psi = \forall\{A_1, B_1\}\ \exists\{E_1, F_1\}\ \forall\{A_2, B_2\}\ \exists\{E_2, F_2\},\ T)$$

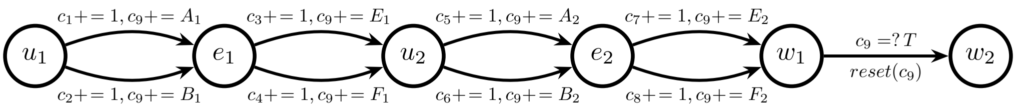
Play table. Recall also that we wish to evaluate whether the following table of plays lead to the sum of the game always being equal to T (which is winning for the existential player).

Play number	u_1	e_1	u_2	e_2	Sum
1	A_1	E_1 or F_1	A_2	E_2 or F_2	$= T$?
2	A_1	*Same as play 1*	B_2	E_2 or F_2	$= T$?
3	B_1	E_1 or F_1	A_2	E_2 or F_2	$= T$?
4	B_1	*Same as play 2*	B_2	E_2 or F_2	$= T$?

We would like to construct a safe counter-stack automata A and compute an initial and target configuration such that if reachability holds, then there is a winning strategy for the existential player. The counter stack automata will have nine counters $c_1, c_2, \ldots, c_9$. The role of the highest order counter, c_9, will be to track the current sum in the game. At all times $c_9 \leq \max\{A_1, B_1\} + \max\{E_1, F_1\} + \max\{A_2, B_2\} + \max\{E_2, F_2\}$. The role of the lower order counters $c_1, \ldots, c_8$ will be to track the decisions made over several of the last plays of the game and to force the enumeration of every play in play table.

The 9-dimensional counter-stack automaton A will consist of two gadgets: the play gadget and the reset gadget.

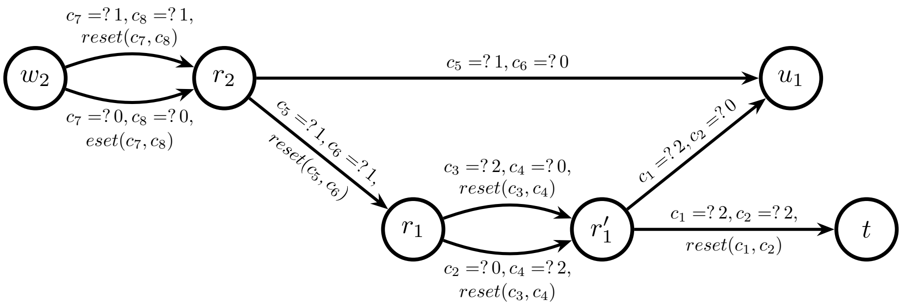
Play gadget. The play gadget allows for an arbitrary play to take place. The counter c_1 is incremented if A_1 is chosen and c_2 is incremented if B_1 is chosen. Then counter c_3 is incremented if E_1 is chosen and c_4 is incremented if F_1 is chosen. The same is true for c_5, c_6, c_7, and c_8 tracking whether A_2, B_2, E_2, and F_2 were chosen, respectively. Throughout the play gadget, c_9 tracks the current sum of the play, and it is only possible to reach the state w_2 (at the end of the play gadget) if c_9 sums to exactly T which corresponds to an individual play being winning for the existential player.



Reset gadget. The reset gadget is designed to control the play gadget; it will only allow plays to take place in a very specific order (if indeed, the final state is reached). This specific order will exactly correspond to the plays presented in the play table. Note that the states w_2 and u_1 are exactly the states w_2 and u_1, respectively, in the play gadget; it is true that these two gadget together form the entire counter-stack automaton A. Recall that in a counter-stack automaton, whenever a counter of index i is equality-tested or reset, then it must be true that the value of all counters of index j for $j > i$ must known (i.e. they can be equality tested as well). In the following picture, whenever a counter c_i is equality-tested, then notice that all lower index counters are also equality-tested at the same time or have recently been reset and have since not been changed and so are known to have value 0. For example, all transitions in zero-test or reset counters of index at most 8, so the

4th December 2025

value of c_9 must be known. Notice that before w_2 is reached in the play gadget, c_9 was reset (by the transition from w_1 to w_2) and the value of c_9 is not changed in the reset gadget, so its value is known to be zero.



Correctness. We will now (roughly) argue that there is a run from $(u_1, \mathbf{0})$ to $(t, \mathbf{0})$ if and only if the existential player has a winning strategy.

First, we shall make some preliminary observations. Notice that once w_2 is reached, and a run enters the reset gadget, then there are three potential outcomes: the run reaches t, the run get stuck and cannot pass some equality test, or the run reaches u_1 and goes through another iteration of passing through the play gadget before re-entering the reset gadget. Notice also that the final transition that leads to t requires that $c_1 = 2$ and $c_2 = 2$ and recall that the only was c_1 or c_2 is increased was by a the first two transitions in the play gadget. In fact, it is only ever possible to increment c_1 or c_2 by 1 in any run through the play gadget. Together, this means that the play gadget must be visited four times before there is a chance that t could be reached. This is good sign as there are four plays to evaluate (see the play table). Also notice that every time w_2 is reached, a play is checked to have sum T (aka. the play is winning for the existential player) by the equality-testing transition between w_1 and w_2.

Now, let us consider what happens after the first play. Notice that once w_2 is reached, the first action of the reset gadget is to reset c_7 and c_8; this corresponds to the fact that in every play in the play table the existential player is *always* allowed to choose between E_2 and F_2 (the reset gadget therefore just erases the record of this decision so it can freely be made again later). Now the run reaches the state r_2. There are two transition leaving r_2: one tests for $c_5 = 1$ and $c_6 = 0$ and the other test for $c_5 = 1$ and $c_6 = 1$. Since it is only possible to increment either c_5 or c_6 by 1 during the first run through the play gadget, in order to not get stuck at r_2, the run must have previously selected the A_2-transition in the play gadget. This means we are iterating over the universal player selecting A_2 in round 2 and this exactly lines up with the first play in the play table.

It is also true that the transition from r_2 does not reset counters c_5 or c_6 (nor does it reset $c_1, c_2, c_3,$ and c_4, so at the start of the second play, we know that $c_5 = 1$ and $c_6 = 0$. Looking ahead, we know that the run will again arrive at r_2. Before arriving at r_2, during the play gadget, the run was able to one again increment c_5 or c_6 by 1. Now, recall that the only two transitions from r_2 test c_5 and c_6: one tests for $c_5 = 1$ and $c_6 = 0$ and the other test for $c_5 = 1$ and $c_6 = 1$. One would therefore become stuck if c_5 was incremented in both the first play and the second play, so this time around c_6 must be incremented. This means we are iterating over the universal player selecting B_2 in round 2 and this exactly lines up with the second play in the play table.

Now we have arrived at r_1. There are two transition from r_1 in the reset gadget: one tests $c_3 = 2$ and $c_4 = 0$ and the other tests $c_3 = 0$ and $c_4 = 0$. Given that in one traversal of the play gadget, it is only possible to increment c_3 or c_4 by one, we know that in order to take either of the transitions from r_1, we must have either chosen to increment c_3 by 1 twice or to increment c_4 by 1 twice. This means that the decision between E_1 and F_1 that was made in the first two plays must be the same. This is exactly what is required by the play table for first and second plays.

Now, suppose we have arrived at r'_1. There are two outgoing transition: one tests $c_1 = 2$ and $c_2 = 0$ (and leads back to the play gadget) and the other tests $c_1 = 2$ and $c_2 = 2$ (and leads to the target state t). Since we have only completed two plays, the only possible transition that one could wish to take is the transition from r'_1 to u_1. In order for this to be successful, during the first and second visits to the play gadget, we must have selected A_1 both times which, again, is exactly what is required by the play table for the for the first and second plays. It is true that the transition from r'_1 to u_1 does not reset c_1 and c_2.

Now, exactly the same arguments can be applied for third and fourth iterations of looping through the play and reset gadgets. The only difference being that at the end of the fourth cycle, once r'_1 is reached for the final time, in order to conclude the run, one must have selected the B_1-transition on both the third and forth iterations through the play gadget. This is exactly what is required by the play table.

Hence, there is a run from $(u_1, \mathbf{0})$ to $(t, \mathbf{0})$ if and only if the existential player has a winning strategy in this two-round subset-sum game.

Reduction time and safeness. Lastly, we will comment on the time required to construct A. In full generality, the automaton has linear size: there are $4n+1$ many counters in the stack, $4n+3$ many states, and less than $8n+6$ many transitions. The automaton A can clearly be constructed in linear time with respect to the size of the of the given subset-sum game (ψ, T).

We know that $c_9 \leq A_1 + B_1 + E_1 + F_1$; more generally, the "sum counter" c_{4n+1} will be bounded above by $\sum_{i=1}^{n} A_i + B_i + E_i + F_i$. It is also true that $c_1, c_2, \ldots, c_8 \leq 3$ because, on every transition in the reset gadget, there is an equality test for $c_1, c_2, \ldots, c_8$ to to check $c_i = 0$, $c_i = 1$, or $c_i = 2$. This means that the play gadget can only be visited at most one more time than this highest equality test, hence $c_1, c_2, \ldots, c_8 \leq 3$. More generally, $c_i \leq 2^{n-1} + 1$ for all the lower order counters c_i that are not the "sum counter". Overall, we therefore know that A is b-safe for $b = \max\{\sum_{i=1}^{n} A_i + B_i + E_i + F_i, 2^{n-1} + 1\}$. ☐

Corollary 9.1. Reachability in binary-encoded 2-VASS is PSPACE-hard.

Proof idea. We shall reduce from reachability in binary-encoded bounded 1-VASS which is PSPACE-hard (Theorem 8.2). Let $(V, (p, x), (q, y))$ be the given instance of reachability in a binary-encoded bounded 1-VASS $V = (Q, T, B)$. We shall construct a 2-VASS $V' = (Q', T')$ that simulates V. Let $c_1 and c_2$ be the two counters of V'; the main idea is to maintain the invariant $c_1 + c_2 = B$ and c_1, c_2 store the value of the counter c in V. The simplest way of encoding this into a 2-VASS is to have $c_1 = c$ and $c_2 = B - c$. Accordingly, we construct the 2-VASS V' to have the same set of states $Q' = Q$ and the transitions are copied to maintain the invariant, so $T' = \{(p, (u, -u), q) : (p, u, q) \in T\}$. It is not difficult to see that V' faithfully simulates V; consider a situation in which a transition (p, u, q) would send a configuration (p, x) to a negative value $x + u < 0$, then c_1 would go negative in V' and, similarly, if (p, u, q) would send a configuration (p, x) above the bound, so $x + u > B$, then c_2 would go negative in V'.

It is true that there is a run from (p, x) to (q, y) in V if and only if there is a run from $(p, (x, B - x)$ to $(q, (y, B - y)$ in V'. ☐

Corollary 9.2. Coverability in binary-encoded 2-VASS is PSPACE-hard.

Proof idea. The idea behind this corollary is almost identical to the one presented in the proof idea of Corollary 9.1. The only difference is in the conclusion. Recall that there is the invariant $c_1 + c_2 = B$. Observe that all transition $(p, (u_1, u_2), q)$ in V' have the property that $u_1 + u_2 = 0$. This means that from the initial configuration $(p, (x, B - x))$ the only reachable configurations $(p', (x_1, x_2)$ satisfy $x_1 + x_2 = B$. Hence it is not possible for V' to reach a configuration $(q, \mathbf{y})$ such that $\mathbf{y} > (y, B - y)$. In other words, for V' reachability and coverability coincide (thanks to the invariant). ☐

Theorem 9.3. Coverability in VASS is in EXPSPACE (regardless of whether the problem is encoded in unary or binary). Moreover, coverability in unary-encoded and binary-encoded d-VASS is in NL and PSPACE, respectively.

To prove Theorem 9.3, we will (in the next lecture) establish the existence "short" runs for coverability. Here "short" means that the runs are doubly exponential in length; i.e. if coverability holds, then there is a run of length at most $n^{d^{\mathcal{O}(d)}}$ (here n is the unary-encoded size of the instance and d is the dimension of the VASS). Given that there are double exponential length runs, then it follows that there is a non-deterministic algorithm for coverability that uses $d^{\mathcal{O}(d)} \cdot \log(n)$ space. This places coverability in VASS in NEXPSPACE, and by Satich's theorem, NEXPSPACE = EXPSPACE. When the dimension d of the VASS is fixed, then $d^{\mathcal{O}(d)}$ is just some (large) constant, so we deduce that there is a non-deterministic $\mathcal{O}(\log(n))$-space algorithm for coverability in unary-encoded d-VASS. So, when the input is encoded in unary, the problem is in NL, and when the input is encoded in binary, the problem is in NPSPACE = PSPACE (again, thanks to Savitch's theorem).