A. Suppose for the sake of contradiction that H decides ACC_{TM}. We describe a decider D for $HALT_{TM}$. On input $\langle M, w\rangle$, D performs this algorithm:

1. Run H on $\langle M, w\rangle$. If H accepts, then accept. Otherwise, continue.
2. Build a Turing machine N that is identical to M except that the accept and reject states are switched.
3. Run H on $\langle N, w\rangle$. If H accepts, then accept. Otherwise, reject.

Because all three steps of D halt, D is a decider. Further,

$$\begin{aligned} D \text{ accepts } \langle M, w\rangle &\Leftrightarrow H \text{ accepts } \langle M, w\rangle \text{ or } H \text{ accepts } \langle N, w\rangle \\ &\Leftrightarrow M \text{ accepts } w \text{ or } N \text{ accepts } w \\ &\Leftrightarrow M \text{ accepts } w \text{ or } M \text{ rejects } w \\ &\Leftrightarrow M \text{ halts on } w \\ &\Leftrightarrow \langle M, w\rangle \in HALT_{TM}. \end{aligned}$$

Thus D is a decider for $HALT_{TM}$. But $HALT_{TM}$ is undecidable. This contradiction implies that ACC_{TM} is also undecidable.

[Can you optimize D down to a single invocation of H? Many students did in their solutions.]

B.A. Yes. Consider a nondeterministic Turing machine with time complexity $t(n)$ and space complexity $s(n)$. Recall that

$$s(n) = \max_{w:|w|=n} \left(\max_{\text{branches on } w} (\text{space usage of that branch}) \right).$$

But, on each branch, the space usage is at most one more than the time usage. Therefore

$$\begin{aligned} s(n) &\leq \max_{w:|w|=n} \left(\max_{\text{branches on } w} (1 + \text{time usage of that branch}) \right) \\ &= 1 + \max_{w:|w|=n} \left(\max_{\text{branches on } w} (\text{time usage of that branch}) \right) \\ &= 1 + t(n). \end{aligned}$$

Thus $s(n)$ is $\mathcal{O}(t(n))$.

B.B. On two occasions we have described a four-tape deterministic Turing machine M for simulating N. The first tape stores the input and hence uses space $n = \mathcal{O}(t_N(n))$ by the assumption that $t_N(n) \geq n$. The second tape simulates N's tape and hence uses space $s_N(n) = \mathcal{O}(t_N(n))$ by the space-time lemma. The third tape stores the branching pattern of N and hence uses space $\mathcal{O}(t_N(n))$. The fourth tape uses space $\mathcal{O}(1)$. Hence $s_M(n)$ is $\mathcal{O}(t_N(n))$. Converting the multi-tape M to a single-tape D increases the space usage by a constant multiple. Hence $s_D(n) = \mathcal{O}(s_M(n)) = \mathcal{O}(t_N(n))$.

[Here's an alternative idea. First, $s_D(n)$ is $\mathcal{O}(t_D(n))$ by the space-time lemma. Second, $t_D(n)$ is $2^{\mathcal{O}(t_N(n))}$. Third, $2^{\mathcal{O}(s_N(n))}$ is $2^{\mathcal{O}(t_N(n))}$ by the space-time lemma. Is there any way to combine these facts to get a bound?]

C.B. This was a homework problem. The language is decidable and hence recognizable and co-recognizable.

C.C. Suppose for the sake of contradiction that the language is decidable by a decider H. Then one can build a decider D for the language $\{\langle M\rangle : M \text{ accepts } \epsilon\}$. (Briefly, D runs H on $\langle M, \epsilon\rangle$. If H rejects, then D rejects. If H accepts, then D runs M on ϵ and outputs whatever M outputs.) But the latter language is undecidable by Rice's theorem. This contradiction shows that the language in question is undecidable. It is recognizable, because a recognizer could simply run M on ϵ and accept whenever M halted. Because the language is undecidable and recognizable, it cannot be co-recognizable.

C.D. For brevity, call the language A. I claim that A is decidable and hence recognizable and co-recognizable. To support this claim, we design a decider D that, on input $\langle M, c\rangle$, does the following computation.

1. For each input w such that $|w| \leq c + 1$:
   (a) Simulate M on w for up to c steps.
   (b) If M does not halt on w in c or fewer steps, then reject.
2. Accept.

This D is a decider because its loop fires finitely many times, doing finitely much work per firing. Now fix an $\langle M, c\rangle$. For brevity, call a string w *short* if $|w| \leq c + 1$. Suppose that D rejects $\langle M, c\rangle$. Then it must have found a w on which M does not halt in c or fewer steps, so $\langle M, c\rangle \notin A$. Conversely, suppose that D accepts $\langle M, c\rangle$. Then M halts on all short w in c or fewer steps. Let x be any input to M such that $|x| > c + 1$, and let w be the string consisting of the first $c + 1$ characters in x. Then M's behavior on input x is identical to M's behavior on input w, because M halts before it can reach the latter characters in x. So, because M halts on all short w in c or fewer steps, it halts on all inputs x in c or fewer steps. So $\langle M, c\rangle \in A$. Thus D decides A.

C.E. The language is undecidable by Rice's theorem. I don't know more than that. So the recognizable and co-recognizable adjectives were removed from grading.

C.F. Any M can be modified into an N that has a disconnected state, simply by adding a state that loops on itself. Therefore the language consists of all Turing machine encodings $\langle M \rangle$. In the jargon of Rice's theorem, it is a trivial property of recognizable languages. It is decidable and hence recognizable and co-recognizable.

C.G. The language is undecidable by Rice's theorem. It is recognizable, because a recognizer could simply run M on $\langle M \rangle$ and output whatever M outputted. Because the language is undecidable and recognizable, it cannot be co-recognizable.

C.H. The language is decidable and hence recognizable and co-recognizable. All of our algorithms for processing such pairs — for example, the recognizer for ACC_{TM} — implicitly perform this check at the start of their computation.