Solution 1

Problem 1.1

Construct Turing Machine as following:

1. Read the rightmost bit of the input. If it is not $\star$, then earse it from the tape and move to the right; otherwise change it to 0, and we are done.
2. When a $\star$ is seen, write 1.
3. Move back to the left, read the rightmost bit of the input.If it is not $\star$, then earse it from the tape and move to the right; otherwise we are done.
4. If 1 is seen, change it to 0 and move right, otherwise change it to 1 and goto 3..

Problem 1.2

Detail omitted.

Problem 1.3

If odd positions are considered to extend to the left and even positions are considered to extend to the right, then unidirectional tape and bidirectional tape are equivalent.

Problem 1.4

$\rightarrow$: If L is recursively enumerable, then there exists a turing machine that halts on every $x \in L$. In case of halting, it outputs 1 if $x \in L$ and outputs 0 otherwise. So we can construct a turing machine T', which simulates T. If T halts and outputs 0, T' comes into an infinite loop. $\leftarrow$: Similarly, we construct a turing machine T which simulates T' and when T halts, it outputs 1.

Problem 1.5

If this proposition is true, we can construct a turing machine which solves the halting problem as follows. For any turing machine T and an input x, there exists a universal turing machine U which simulates T on the input x. Then by the proposition, there exists another turing machine U' and U' halts on (Tx) if and only if U does not halt on (T, x). So that we can construct a new turing machine D, which simulates U and U' on the input (T, x) in parallelism. When U halts, D halts and outputs that T halts on input x, and when U' halts D halts and outputs that T does not halt on input x. Thus, D can solve the halting problem, which is impossible.

Problem 1.6

No. Considering a universal turing machine U, we can find a turing machine T^* and an input x^* where T halts on x. So that $f_U(T^*, x^*) = 1$. If the language is recursive, then there exist a turing machine D which decides $f_U(T, x) = 1 or 0$ on every T and x. This solves the halting problem and is impossible.

Problem 1.7

(a) Yes. We can construct a Turing machine U which simulates T and S on the input x simultaneously, and if T halts before S, it outputs 1, otherwise outputs 0. $\forall(T, S, x)$ in this language, we know T must halt, so U must halt too. When U halts, it outputs 1 if (T, S, x) is in this language.

(b) No. If it is true, there exists a Turing machine D which halts on every (T, S, x) in this language, then we can construct a Turing machine U which decides the halting problem. On an input (S, x), U invokes D with (T, S, x) for a T which does not halt on x, and simulates S simultaneously. If S does not halt, D finally halts, because T is as fast as S, so that U halts. And if S halts, U halts too.