2.

This is true. Indeed, consider such a pair (m, w) and consider a perfectmatching containing pairs (m, w') and (m', w), and, hence, not (m, w). Then since m and w each rank the other first, they each prefer the other to their partners in this matching, and so this matching cannot be stable.

3.

There is not always a stable pair of schedules. Suppose Network $\mathcal{A}$ has two shows $\{a_1, a_2\}$ with ratings 20 and 40; and Network $\mathcal{D}$ has two shows $\{d_1, d_2\}$ with ratings 10 and 30.

Each network can reveal one of two schedules. If in the resulting pair, a_1 is paired against d_1, then Network $\mathcal{D}$ will want to switch the order of the shows in its schedule (so that it will win one slot rather than none). If in the resulting pair, a_1 is paired against d_2, then Network $\mathcal{A}$ will want to switch the order of the shows in its schedule (so that it will win two slots rather than one).

5.

(a) The answer is Yes. A simple way to think about it is to break the ties in some fashion and then run the stable matching algorithm on the resulting preference lists. We can for example break the ties lexicographically that is if a man m is indifferent between two women w_i and w_j then w_i appears on m's preference list before w_j if $i < j$ and if $j < i$ w_j appears before w_i. Similarly if w is indifferent between two men m_i and m_j then m_i appears on w's preference list before m_j if $i < j$ and if $j < i$ m_j appears before m_i.

Now that we have concrete preference lists, we run the stable matching algorithm. We claim that the matching produced would have no strong instability. But this latter claim is true because any strong instability would be an instability for the match produced by the algorithm, yet we know that the algorithm produced a stable matching — a matching with no instabilities.

(b) The answer is No. The following is a simple counterexample. Let $n = 2$ and m_1, m_2 be the two men, and w_1, w_2 the two women. Let m_1 be indifferent between w_1 and w_2 and let both of the women prefer m_1 to m_2. The choices of m_2 are insignificant. There is no matching without weak stability in this example, since regardless of who was matched with m_1, the other woman together with m_1 would form a weak instability.

8.

Assume we have three men m_1 to m_3 and three women w_1 to w_3 with preferences as given in the table below. Column w_3 shows true preferences of woman w_3, while in column w_3' she pretends she prefers man m_3 to m_1.

m_1	m_2	m_3	w_1	w_2	w_3	(w_3')
w_3	w_1	w_3	m_1	m_1	m_2	m_2
w_1	w_3	w_1	m_2	m_2	m_1	m_3
w_2	w_2	w_2	m_3	m_3	m_3	m_1

First let us consider one possible execution of the G-S algorithm with the true preference list of w_3.

m_1	w_3			w_3
m_2		w_1		w_1
m_3			$[w_3][w_1]w_2$	w_2

First m_1 proposes to w_3, then m_2 proposes to w_1. Then m_3 proposes to w_2 and w_1 and gets rejected, finally proposes to w_2 and is accepted. This execution forms pairs (m_1, w_3), (m_2, w_1) and (m_3, w_2), thus pairing w_3 with m_1, who is her second choice.

Now consider execution of the G-S algorithm when w_3 pretends she prefers m_3 to m_1 (see column w_3'). Then the execution might look as follows:

m_1	w_3		$-$	w_1			w_1
m_2		w_1		$-$	w_3		w_3
m_3			w_3		$-$	$[w_1]w_2$	w_2

Man m_1 proposes to w_3, m_2 to w_1, then m_3 to w_3. She accepts the proposal, leaving m_1 alone. Then m_1 proposes to w_1 which causes w_1 to leave her current partner m_2, who consequently proposes to w_3 (and that is exactly what w_3 wants). Finally, the algorithm pairs up m_3 (recently left by w_3) and w_2. As we see, w_3 ends up with the man m_2, who is her true favorite. Thus we conclude that by falsely switching order of her preferences, a woman may be able to get a more desirable partner in the G-S algorithm.