Since we are interested in a maximal $F$-set, the computation can be expressed:

\[
\begin{align*}
&\left( (b \lor \varphi) \diamond \neg (d \lor \varphi) \land d_+ \lor \left( (f \lor \varphi) \diamond \neg (d \lor \varphi) \lor \neg \varphi \land \neg \varphi \land \varphi \right) \right) \\
&\quad \land \varphi \\
&\quad \cap \neg \varphi \\
&\quad \varphi
\end{align*}
\]

This can be summarized as:

- For every $c \in (b \lor \varphi) \diamond \neg (d \lor \varphi)$ and $d_+$.
- For every $c \in (f \lor \varphi) \diamond \neg (d \lor \varphi)$.

Every state has a successor.

F-sets.

Computing F-sets.

Assume an assertion $\varphi$ characterizes an $F$-set. Translating the requirements into assertions:

Satisfying:

- $b \lor \varphi$.
- $d \lor \varphi$.

Claim: $F$-sets.

Proof:

There exists at least one computation from some state to some state or satisfies.

For every state $s \in S$ and every $F$-set.

There exists a successor.

For every state $s \in S$.

Each state has a successor.

All states in $S$ are reachable.

A set of states is defined to be an $F$-set if it satisfies the following:

1. The state is reachable.
2. For any state $s$ in $S$ and every $F$-set.
3. The state is reachable.
4. Each state is reachable.

Feasible:

A state is feasible if it can be included in some run. The $F$-sets are called feasible if they are reachable.

Checking for feasibility:

A state in $S$ is called feasible if its requirements are satisfied by any of the requirements.

A run in $P$ is an initial sequence of states which satisfies the requirements.

We assume that a feasible state is reachable.

All states in $S$ are reachable.

F-sets.

A property is called feasible if it is reachable.

A property is called feasible if it can be included in some run.

A property is called reachable if its requirements are satisfied by any of the requirements.

A run in $P$ is an initial sequence of states which satisfies the requirements.

A state in $S$ is called feasible if its requirements are satisfied by any of the requirements.

A set of states is defined to be an $F$-set if it satisfies the following:

1. The state is reachable.
2. For any state $s$ in $S$ and every $F$-set.
3. The state is reachable.
4. Each state is reachable.

Feasible:

A state is feasible if it can be included in some run. The $F$-sets are called feasible if they are reachable.

Checking for feasibility:

A state in $S$ is called feasible if its requirements are satisfied by any of the requirements.

A run in $P$ is an initial sequence of states which satisfies the requirements.
Algorithmic Interpretation

Computing the maximal fix-point as a sequence of iterations, we can describe the computational process as follows:

- Start by letting $V = \text{reachable}$. Then repeat the following steps:
  - For each $j \in J$, remove from $V$ all states which do not have a $j$-successor.
  - Remove from $V$ all states which do not have a $q$-successor.
  - For each $(p, q) \in E$, remove from $V$ all states which do not have a $p\to q$-successor.

To check whether an FDS $D$ is feasible, we compute for it the maximal $F$-set and check whether it is empty. $D$ is feasible if the maximal $F$-set is not-empty.

Let $D \equiv (V, E, f, p, J, C)$ be an FDS. $D$ has a feasible response property if and only if $D$ is feasible and $f \equiv g$, where $g$ is the transition relation that every computation of $D$ which violates $p$ must avoid.

**Claim: Model Checking Response.**

$D$ is feasible and $f \equiv g$.

**Proof:**

The claim is justified by the observation that every computation of $D$ must avoid all $p\to q$-successors, which are the states in $C$.

First, we eliminate all $(T_2, 1) = 1$-states which do not have a path leading to a $C$-state. This leaves us with:

Assume we wish to verify the property $p \Rightarrow C_2$.

We start by forming $D$ by setting $p = \emptyset$ and then proceed as follows:

- Removing $h \equiv \{0\}$ and then proceed as follows:
  - Removing from $V$ all states which do not have a $q$-successor.
  - Removing from $V$ all states without successors, leaves $\phi_0 : \{3, 4, 5\}$.
  - Successively removing from $V$ all states which do not have a $q$-successor.
  - Removing from $V$ all states without successors.

We conclude that the above FDS is feasible.

Following is the set of all reachable states of program MUX-SEM:

Let $D \equiv (V, E, f, p, J, C)$ be an FDS and $p \Rightarrow g$ be a response property we wish to verify over $D$. Let $\text{reachable}$ be the assertion characterizing all the reachable states in $D$.

We define an auxiliary FDS $D' \equiv (V, E, f, p\land q, J, C)$, where $q$ allows any $p$-step as long as the successor does not satisfy $q$.

Thus, $\phi_q$ characterizes all the $r$-reachable states which do not satisfy $q$.

### Example

As an example, consider the following FDS:

```
As an example, consider the following FDS:

- Start by setting $V = \text{reachable}$. Then repeat the following steps:
  - For each $j \in J$, remove from $V$ all states which do not have a $j$-successor.
  - Remove from $V$ all states without successors.

To check whether an FDS $D$ is feasible, we compute for it the maximal $F$-set and check whether it is empty. $D$ is feasible if the maximal $F$-set is not-empty.

Let $D \equiv (V, E, f, p, J, C)$ be an FDS and $p \Rightarrow g$ be a response property we wish to verify over $D$. Let $\text{reachable}$ be the assertion characterizing all the reachable states in $D$.

We define an auxiliary FDS $D' \equiv (V, E, f, p\land q, J, C)$, where $q$ allows any $p$-step as long as the successor does not satisfy $q$.

Thus, $\phi_q$ characterizes all the $r$-reachable states which do not satisfy $q$.

### Example: MUX-SEM

Following is the set of all reachable states of program MUX-SEM:

Let $D \equiv (V, E, f, p, J, C)$ be an FDS and $p \Rightarrow g$ be a response property we wish to verify over $D$. Let $\text{reachable}$ be the assertion characterizing all the reachable states in $D$.

We define an auxiliary FDS $D' \equiv (V, E, f, p\land q, J, C)$, where $q$ allows any $p$-step as long as the successor does not satisfy $q$.

Thus, $\phi_q$ characterizes all the $r$-reachable states which do not satisfy $q$.

### Example: MUX-SEM

Following is the set of all reachable states of program MUX-SEM:

Let $D \equiv (V, E, f, p, J, C)$ be an FDS and $p \Rightarrow g$ be a response property we wish to verify over $D$. Let $\text{reachable}$ be the assertion characterizing all the reachable states in $D$.

We define an auxiliary FDS $D' \equiv (V, E, f, p\land q, J, C)$, where $q$ allows any $p$-step as long as the successor does not satisfy $q$.

Thus, $\phi_q$ characterizes all the $r$-reachable states which do not satisfy $q$.
Next, we eliminate all states which do not have a path leading to a $C^1$-state. This leaves us with nothing. We conclude that $\text{MUX-sem} = T_2$. 

Analysis of Reactive Systems, NYU, Fall, 2009