1.2 Application: System Specifications

Having hardware or software specification in natural language we would like to be able to translate them into logical expressions. This will make the specification *precise* and *unambiguous*, and can be used as basis for system development.

System specification is consistent if and only if it doesn't have any *conflicting requirements* that could be used to derive a *contradiction*.

When specifications are not consistent, there would be no way to develop a system that satisfies all specifications.

Example 3 (page 20/50): Determine whether the system specification is consistent (i.e. not contradictory)

“Whenever the system software is being upgraded, users cannot access the file system. If users can access the file system, then they can save new files. If users cannot save new files, then the system software is not being upgraded”

System Specifications

System specification is consistent if and only if it doesn't have any *conflicting requirements* that could be used to derive a *contradiction*.

When specifications are not consistent, there would be no way to develop a system that satisfies all specifications.

Example 3 (page 22/10): Determine whether the system specification is consistent (i.e. not contradictory)

“Whenever the system software is being upgraded, users cannot access the file system. If users can access the file system, then they can save new files. If users cannot save new files, then the system software is not being upgraded”

p: “the system software is being upgraded”
q: “users can access the file system”
r: “users can save new files”

$$\begin{cases} p \to \neg q \\ q \to r \\ \neg r \to \neg p \end{cases}$$

let's check if there exists an assignment of truth values (to p, q, and r) that makes all of these compound propositions true.

System Specifications

Example 4 (page 22/9): Determine whether the system specification is consistent.

“The system is in multiuser state if and only if it is operating normally. If the system is operating normally the kernel is functioning.
The kernel is not functioning or the system is in interrupt mode. If the system is not in multiuser state, then it is in interrupt mode. The system is not in interrupt mode.”

p: “the system is in multiuser state”
q: “it is operating normally”
r: “the kernel is functioning”
s: “the system is in interrupt mode”

$$\begin{cases} p \leftrightarrow q \\ q \rightarrow r \\ \neg r \vee s \\ \neg p \rightarrow s \\ \neg s \end{cases}$$

same technique:
let's check if there exists an assignment of truth values (to p, q, r, and s) that makes all of these compound propositions true.

System Specifications

Example 4 (page 22/9): Determine whether the system specification is consistent.

$$\begin{cases} p \leftrightarrow q \\ q \rightarrow r \\ \neg r \vee s \\ \neg p \rightarrow s \\ \neg s \end{cases}$$

same technique:
let's check if there exists an assignment of truth values (to p, q, r, and s) that makes all of these compound propositions true.

1) $\neg s$ must be True, therefore s is False
2) if s is False, then in order for implication $\neg p \rightarrow s$ to be True, $\neg p$ must be also False. Therefore, p is True.
3) if p is True, then in order for $p \leftrightarrow q$ to be True, q must be True.
4) if q is True then in order for $q \rightarrow r$ to be True, r also must be True.
5) if r is True, and s is False, then $\neg r \vee s$ is False $\vee$ False, which gives us False.

So, there is no assignment of truth values that makes all of the above propositions true. Hence, the system specification is inconsistent.