0. Introduction

We present here a sound and complete axiomatization of event structures. Event structures are a poset-based model of distributed systems. In an event structure, the three fundamental phenomena of causality, choice and concurrency are clearly separated from each other.

Event structures are pleasant mathematical objects possessing a rich theory [NPW]. They have a very natural connection to the theory of Petri nets [NPW]. Event structures can be used to provide the non-interleaved denotational semantics of a large family of CCS-like languages [W1]. Hence there is a good deal of motivation for developing a logic to reason about the behaviour of a distributed system represented in terms of an event structure. The logic we present here is a temporal logic with two additional unary modalities which directly express choice and concurrency.

In the next section we introduce event structures. In Section 2 we define the logical language and specify a Kripke-style semantics using event structures as frames. We then illustrate with the help of some simple examples how this language can be used to express interesting properties of distributed computations. The axiomatization is presented in Section 3. In Section 4, we sketch the proof of completeness of the axiomatization, which is the main contribution of this paper. The concluding section discusses related work and directions for future research.

1. Event Structures

An event structure essentially consists of a partially ordered set of event occurrences augmented with a binary conflict relation.
Definition 1.1:
An event structure is a triple \( \text{ES} = (E, <, \#) \) where

(i) \( E \) is a set of events.
(ii) \( < \subseteq E \times E \) is an irreflexive and transitive causality relation.
(iii) \( \# \subseteq E \times E \) is an irreflexive and symmetric conflict relation.
(iv) \( \# \) is inherited via \( < \) in the sense that \( e_1 \# e_2 < e_3 \) implies that \( e_1 \# e_3 \) for every \( e_1, e_2, e_3 \) in \( E \).

To be precise, the objects we have defined above are usually referred to as prime even% structures in the literature [W2]. Here, for the sake of convenience, we always refer to them simply as event structures. Usually the causality relation is required to be a partial ordering relation. We have made it a strict partial ordering relation because it fits in better with the completeness argument presented in Section 4.

Let \( \text{ES} = (E, <, \#) \) be an event structure. Then

\[
\begin{align*}
\text{id} & \triangleq \{(e, e) \mid e \in E\} \\
\triangleright & \triangleq \{(e, e') \mid (e', e) \in <\} \\
\leq & \triangleq \# \cup \text{id} \\
\geq & \triangleq \triangleright \cup \text{id} \text{ and} \\
\bowtie & \triangleq E \times E - (\leq \cup \geq \cup \#).
\end{align*}
\]

The relation \( \bowtie \) captures concurrency. Observe that \( \{<, \triangleright, \#, \bowtie, \text{id}\} \) is a partitioning of \( E \times E \).

Before we consider an example it will be convenient to define one more auxiliary relation. Let \( \text{ES} = (E, <, \#) \) be an event structure and \( e, e' \in E \). Then

\[
\begin{align*}
\#_\mu & \triangleq e \# e' \text{ and} \\
\forall e_i, e'_i & \in E. [ e_i \leq e \text{ and } e_i' \leq e' \text{ and } e_i \# e_i' \text{ implies} \\
e_i = e \text{ and } e_i' = e'].
\end{align*}
\]

\( \#_\mu \) in some sense identifies the "minimal elements" (under \( < \)) of the \( \# \) relation. As a result, we can specify an event structure by displaying its \( < \) and \( \#_\mu \) relations. The \( \# \) relation is then uniquely determined by part (iv) of Definition 1.1.

Figure 1.1 is an example of an event structure. The squiggly lines represent the \( \#_\mu \) relation. The causality relation is shown in the form of the associated Hasse diagram.

In this event structure, \( e_4 \# e_6 \) because \( e_4 \#_\mu e_2 < e_6 \). It is also easy to see that \( e_6 \bowtie e_7 \).