An Inference Mechanism for Point-Interval Logic

Mashhood Ishaque\textsuperscript{1} Faisal Mansoor\textsuperscript{2} Abbas K. Zaidi\textsuperscript{2}

\textsuperscript{1}Tufts University
\textsuperscript{2}George Mason University

Florida Artificial Intelligence Research Society Conference
Formal Logic

- Logic is the “art of reasoning”.
- Logic is used to make inferences based on the available information.
- Formal logic makes inferences based purely on the form of the content, without any understanding of the meaning of the content.
- Reasoning based just on the form is important because this means computers can do it.
We need a logic to reason about temporal information so as to:

- characterize time-sensitive attributes of a domain to be modeled.
- do temporal analysis of a domain, which will help us in developing a better understanding of the relationship between domain entities.
- identify inconsistencies and anomalies.
Applications

- Forensics: Temporal information is the facts (may be partial) about some incident that has already occurred.
- Planning and Scheduling: Temporal information is the constraints/specification that need to imposed on the desired schedule.
Point-Interval Logic

Point-Interval Logic is a temporal logic.

- It is a tractable subclass of Allen’s interval logic.
- Point-Interval Logic (PIL) is a Pointisable logic.
- It combines qualitative and quantitative temporal information.
Point-Interval Logic is a temporal logic.
- It is a tractable subclass of Allen’s interval logic.
- Point-Interval Logic (PIL) is a Pointisable logic.
- It combines qualitative and quantitative temporal information.
Point-Interval Logic is a temporal logic.

- It is a tractable subclass of Allen’s interval logic.
- Point-Interval Logic (PIL) is a Pointisable logic.
- It combines qualitative and quantitative temporal information.
Point-Interval Logic

Point-Interval Logic is a temporal logic.
- It is a tractable subclass of Allen’s interval logic.
- Point-Interval Logic (PIL) is a Pointisable logic.
- It combines qualitative and quantitative temporal information.
**Point Graphs**

**Figure:** Point Graph for a Set of PIL Statements

Point A, B, C
A < B
B ≤ C
Stamp (A) = 100
Length (A, B) = 5
Qualitative Constraints

Interval X, Y

- $sX \rightarrow eX \rightarrow sY \rightarrow eY$

- $X < Y$
  - $sX \rightarrow eX \rightarrow sY \rightarrow eY$  \hspace{1cm} $X$ before $Y$

- $X m Y$
  - $sX \rightarrow eX; sY \rightarrow eY$  \hspace{1cm} $X$ meets $Y$

- $X o Y$
  - $sX \rightarrow sY \rightarrow eX \rightarrow eY$  \hspace{1cm} $X$ overlaps $Y$

- $X s Y$
  - $sX; sY \rightarrow eX \rightarrow eY$  \hspace{1cm} $X$ starts $Y$
Quantitative Constraints

<table>
<thead>
<tr>
<th>Condition</th>
<th>Diagram</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stamp ([P] = t)</td>
<td>![Diagram 1]</td>
</tr>
<tr>
<td>Length ([X] = d)</td>
<td>![Diagram 2]</td>
</tr>
<tr>
<td>Stamp ([P] \geq t)</td>
<td>![Diagram 3]</td>
</tr>
<tr>
<td>Stamp ([P] \leq t)</td>
<td>![Diagram 4]</td>
</tr>
<tr>
<td>Length ([X] \geq d)</td>
<td>![Diagram 5]</td>
</tr>
<tr>
<td>Length ([X] \leq d)</td>
<td>![Diagram 6]</td>
</tr>
</tbody>
</table>
Point-Interval Logic
Inference Mechanism
General Temporal Problem

Point Graph Construction

a) Convert each PIL statement into a Point Graph

b) Unify individual Point Graphs into a single Point Graph

Fold branch and join nodes. Before we fold the Point Graph, we must check it for consistency.

c) Fold the branch nodes (outdegree > 1) in the Point Graph

d) Fold the join nodes (indegree > 1) in the Point Graph

Once we have a folded and consistent Point Graph, we can use it to draw inferences.

Interval X, Y, Z
X o Y
Length (sX, sY) = 10
Length (sY, eX) = 8
Z o Y
Length (sZ, sY) = 5
Length (sY, eZ) = 8

b) Unify the nodes with same labels.
Basic Temporal Queries

An inference mechanism for Point-Interval logic should be able to answer following basic queries:

- Relationship Query
- Stamp Query
- Length Query
Problem with Previous Mechanism

Figure: Illustration of Incompleteness
We only need to infer relationship between points.

<table>
<thead>
<tr>
<th>(sX, sY)</th>
<th>(sX, eY)</th>
<th>(eX, sY)</th>
<th>(eX, eY)</th>
<th>(X, Y)</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;</td>
<td>&lt;</td>
<td>&lt;</td>
<td>&lt;</td>
<td>{&lt;}</td>
</tr>
<tr>
<td>&lt;</td>
<td>&lt;</td>
<td>=</td>
<td>&lt;</td>
<td>{m}</td>
</tr>
<tr>
<td>&lt;</td>
<td>&lt;</td>
<td>&gt;</td>
<td>&lt;</td>
<td>{o}</td>
</tr>
<tr>
<td>&lt;</td>
<td>&lt;</td>
<td>&gt;</td>
<td>=</td>
<td>{f_i}</td>
</tr>
<tr>
<td>&lt;</td>
<td>&lt;</td>
<td>&gt;</td>
<td>&gt;</td>
<td>{d_i}</td>
</tr>
<tr>
<td>&lt;</td>
<td>&lt;</td>
<td>&gt;</td>
<td>?</td>
<td>{o, f_i}</td>
</tr>
<tr>
<td>&lt;</td>
<td>&lt;</td>
<td>&gt;</td>
<td>?</td>
<td>{d_i, f_i}</td>
</tr>
<tr>
<td>&lt;</td>
<td>&lt;</td>
<td>&gt;</td>
<td>?</td>
<td>{o, d_i, f_i}</td>
</tr>
</tbody>
</table>
The longest path from B to D has length at least 10 units, while the longest path from A to D is exactly 5 units. Thus we conclude that B < A.

The longest path (backward) from A to D is exactly 10 units. While the longest path from B to D is at least 15 units. Thus we conclude that A < B.

The longest path from A to D has length at least 5 units, while the longest path (backward) from B to D is at least 10 units. Thus we conclude that A < B and greatest lower bound on path length is 15.

Figure: Inferring Relationship Between Points A and B
Algorithm 1 queryRelation \((p, q)\)

Find the longest paths from \(p\) and \(q\) to every reachable node in the directed graph.
Find the longest paths from \(p\) and \(q\) to every reachable node in the directed graph obtained by reversing the direction of every edge.
Store at each node in the graph the information about the length and direction (forward/reverse) of the longest paths.

\(\text{pathLength}_{gb}(p, q) \leftarrow 0\)

\(\text{for all nodes } v \text{ in the graph do}\)

\(\text{if forwardPath}(p, v) \text{ and backwardPath}(q, v) \text{ then}\)

\(\text{relation}(p, q) \leftarrow '<'\)

\(\text{pathLength}_{ib}(p, q) \leftarrow \text{pathLength}_{gb}(p, v) + \text{pathLength}_{gb}(v, q)\)

\(\text{else if forwardPath}(p, v) \text{ and backwardPath}(q, v) \text{ then}\)

\(\text{if } \text{pathLength}_{gb}(p, v) > \text{pathLength}_{gb}(q, v) \text{ then}\)

\(\text{relation}(p, q) \leftarrow '<'\)

\(\text{pathLength}_{ib}(p, q) \leftarrow \text{pathLength}_{gb}(p, v) - \text{pathLength}_{gb}(q, v)\)

\(\text{end if}\)

\(\text{end if}\)

\(\text{end for}\)
Stamp Query

Greatest lower bound on the length of the path from B to A = 5
Stamp(B) cannot be inferred. but Stamp(B) is less than 95

Figure: Inferring Stamp of the Point B
Algorithm 3 queryStamp (p)

Find the nodes s and t with smallest and largest time stamp respectively using breadth first search.

\( \text{Stamp}(p) \leftarrow ?, \, \text{Stamp}_{\text{glb}}(p) \leftarrow ?, \, \text{Stamp}_{\text{lub}}(p) \leftarrow ? \)

queryRelation \((p, s)\)

if queryRelation returns \(p > s\) then

\(\text{Stamp}_{\text{glb}}(p) \leftarrow \text{Stamp}(s) + \text{pathLength}_{\text{glb}}(s, p)\)

if \(\text{pathLength}_{\text{glb}}(s, p)\) is exact then

\(\text{Stamp}(p) \leftarrow \text{Stamp}_{\text{glb}}(p)\)

return

end if

else

\(\text{Stamp}_{\text{glb}}(p) \leftarrow \text{Stamp}(s) + \text{pathLength}_{\text{glb}}(s, p)\)

return

end if

queryRelation \((p, t)\)

if queryRelation returns \(p < t\) then

\(\text{ Stamp}_{\text{lub}}(p) \leftarrow \text{Stamp}(t) - \text{pathLength}_{\text{glb}}(p, t)\)

end if
Length Query

E and D are the two nodes reachable from both A and B. Based on paths to E we cannot infer anything. But node D allows us to conclude that length of path from B to A is at least 5 units.

**Figure:** Inferring Length of the Interval [B, A]
Generalized Point-Interval Logic

Interval A, B, C
(A \land B) or (A \lor B) or (A \diamond B)
(C < B) or (C > B) (disjointedness constraint)
(C \cap A) or (C \cap A)
(A < D) or (A > D)

Table: An Instance of GPIL
Search Space

Point-Interval Logic
Inference Mechanism
General Temporal Problem

Figure: Search Space Exploration

FLAIRS 2008
An Inference Mechanism for Point-Interval Logic
CMI Algorithm

Algorithm 4 CMI \((S, i)\)

\{The algorithm is invoked by calling CMI \((S, 1)\), and returns true if the instance is satisfiable\}
\((X, Y) \leftarrow \text{Variables in } s_i\)
\(R_1 \leftarrow \text{Relations in } s_i\)
\(R_2 \leftarrow \text{queryRelation } (X, Y)\)
\(\text{sort } (R_1 \cap R_2)\)
\{sort according to desired heuristics\}
\(\text{for all relation } r_j \epsilon (R_1 \cap R_2) \text{ do}\)
\(\text{addStatement } (X, r_j, Y)\)
\(\text{if } i = n \text{ then}\)
\(\quad \text{return true}\)
\(\text{else if CMI } (S, i + 1) \text{ then}\)
\(\quad \text{return true}\)
\(\text{else}\)
\(\quad \text{deleteStatement } (X, r_j, Y)\)
\(\text{end if}\)
\(\text{end for}\)
\(\text{return false}\)
Questions?

Thank you for listening.