Chapter 7

Abductive Reasoning with the Integrative Knowledge Base

This chapter is concerned with extensions of the abductive inference procedure implemented in the reasoning system Mini-TACITUS (Mulkar et al., 2007). The extensions are intended to make the system able to reason with the developed integrative knowledge base.

Since Mini-TACITUS was not originally designed for large-scale processing, it was necessary to perform several optimization steps, so that the system could treat a large knowledge base. The performed optimization steps are described in Sec. 7.1 of this chapter. Section 7.2 focuses on the solutions to two pragmatic problems encountered in the application of weighted abduction to NLU. Sections 7.3 and 7.4 focus on the extensions of Mini-TACITUS enabling the integration of reasoning with ontologies and similarity spaces into the abductive inference procedure.

7.1 Adapting Mini-TACITUS to a Large Knowledge Base

Mini-TACITUS (Mulkar et al., 2007) began as a simple backchaining theorem-prover intended to be a more transparent version of the original TACITUS system, which was based on Stickel’s PTTP system (Stickel, 1988). Originally, Mini-TACITUS was not designed for treating large amounts of data. A clear and clean reasoning procedure rather than efficiency was in the focus of its developers. For making the system work with a large knowledge base, several optimization steps had to be performed and a couple of new features had to be added.¹

¹We thank Niloofar Montazeri, Rutu Mulkar-Mehta, and Jerry Hobbs for their extensive help with the optimization of the Mini-TACITUS system.
For controlling the reasoning complexity problem, two parameters were introduced. The time parameter $t$ is used to restrict the processing time. If the processing time exceeds $t$, reasoning terminates and the best interpretation so far is output. The time parameter ensures that an interpretation will be always returned by the procedure even if reasoning could not be completed in a reasonable time.

The depth parameter $d$ restricts the depth of the inference chain. Suppose a proposition $p$ occurring in the input was backchained upon and a proposition $p'$ was introduced as the result. Then, $p'$ will be backchained upon and so on. The number of such iterations cannot exceed $d$. The depth parameter reduces the number of reasoning steps. Suppose the knowledge base contains the axioms $\text{dog}(e, x) \rightarrow \text{mammal}(e, x)$, $\text{mammal}(e, x) \rightarrow \text{animal}(e, x)$, $\text{animal}(e, x) \rightarrow \text{organism}(e, x)$ and the proposition $\text{organism}(e, x)$ occurs in the input logical form. Given the depth parameter value equal to 2, $\text{organism}$ can be backchained on to $\text{animal}$ (first step) and $\text{mammal}$ (second step), but not to $\text{dog}$.

The interaction between the time and depth parameters is shown in Algorithm 7.1. A logical form $LF$, a knowledge base $KB$, a depth parameter $D$, a default cost parameter $C$, and a time parameter $T$ are input to the algorithm. Propositions from $LF$, with assigned initial default costs $C$ and depth equal to 0, constitute the initial interpretation $I_{init}$ in the interpretation set $I_{set}$. Then, the recursive subroutine $\text{apply\_inference}$ is called, so that the initial interpretation $I_{init}$ with assigned costs and depth is passed to the subroutine as a parameter.

As long as the processing time does not exceed $T$, the subroutine $\text{apply\_inference}$ works as follows. For each axiom $\alpha$ from the knowledge base $KB$, for each subset $PS$ of propositions belonging to the input interpretation $I$ such that these propositions have a depth smaller than the depth parameter $D$ and $\alpha$ can be applied to $PS$, the procedure constructs a new interpretation $I_{new}$ as the result of the application of $\alpha$ to $PS$. The depth of all propositions from $I$, which occur in $I_{new}$, is increased by 1. The new interpretation $I_{new}$ is added to the interpretation set $I_{set}$. The subroutine $\text{apply\_inference}$ is called again, with the parameter $I_{new}$.

After the subroutine $\text{apply\_inference}$ terminates, a set of interpretations having the lowest cost ($\text{Cheapest\_I}$) is selected from the overall set $I_{set}$ of constructed interpretations. Those interpretations from the $\text{Cheapest\_I}$ set, which have the shortest proof length (the