CANDID Specification of Commercial and Financial Contracts: A Formal Semantics Approach to Knowledge Representation, Part I: Syntax & Formal Semantics of CANDID

Ronald M. Lee

Florida International University (FIU), Miami, FL, USA, r.lee@fiu.edu

Abstract. The formal language CANDID is presented as a knowledge representation formalism for artificially intelligent decision support systems. The language is specifically oriented to representation of concepts in finance, commerce and administration. Later parts of the paper demonstrate the application of CANDID to explication of corporate entities and contractual objects, as well as to various concepts in elementary finance.

1 Introduction

This part presents the syntax and formal semantics of the language we have called CANDID, originally described in Lee [Lee80].

In the discussion that follows, the reader is presumed to be familiar with the first order predicate calculus (FOPC), which we takes as our starting point. For background, we suggest the text by Kalish, Mantague and Mar [KMM80]. The extension to this that comprise CANDID are drawn chiefly from Montague's "intensional logic" [Mon02,Dow78], and Von Wright's "deontic logic",[1] with minor influence from the temporal logic of Rescher and Urquhart [RU71]. The presentation given here is a model theoretical one. Background on model theory is given in Dowty [Dow78] Kalish et al. [KMM80] mentioned above. Deeper coverage is provided in van Fraasen [vF71] and Chang and Keisler [CK73].

The CANDID language as described here loosely follows the develop of Montague's Intensional Logic, as presented in Dowty [Dow78], augmented with the operators of Von Wright's Deontic Logic. The principle differences up to the language IL (Intensional Logic) are as follows:

- addition of operators and the definite reference operator ι.
- omission of the tense, P and F (past and future)
- addition of the sets C (character strings) and N (numbers) in the model.

[1] [VW65,VW67,VW68]

- recognition of time (designated as the set T rather than J) with the object language; addition of the operator R for temporal realization (adapted from a similar notation by Rescher and Urquhart.[2]

The language IL is then extended to include the connectives and operators of Von Wright's deontic logic with the following modifications:

- addition of an agent place in the I connective
- re-interpretation of contingent permission and obligation
- addition of operators for contractual obligation and permission, and the connective OE ("or else").

1.1 General Notational Conventions

Throughout this paper we will describe a series of formal languages of increasing complexity. The formal language itself will be called the *object language*, whereas its description is done via a *meta-language.*

Object Language – Constants In the object language, constant names will be strings of upper or lower case Roman letters or digits or dashes, beginning with a capital letter. These will denote individuals in the domain. Later, the object language is extended to include symbolic entities, i.e. character strings and numbers. These may be designated directly in the language, with the intermediate device of a constant name. Character string constants will be shown between double quotes, e.g., "this is a character string", while numeric constants will have the usual Arabic notation, with an optional embedded decimal point, e.g., 1, 4, 5, 98.6. For consistency, these designations will be treated as names for themselves. Thus, the general notation for constants is that they begin with a capital letter, digit, or double quote.

Object Language – Variables Variables will be denoted as one or more lower case letters, with an optional subscript, e.g., x, y, z1, z2.

Meta-Language In the meta-language, constants will be represented using the Greek characters, α, β, γ, Φ, Ψ. Variables will be designated in the meta-language by the characters μ and ν.

[2] [RU71]