Talking to Godot: Dialogue with a Mobile Robot

Christian Theobalt, Johan Bos, Tim Chapman, Arturo Espinosa-Romero,
Mark Fraser, Gillian Hayes, Ewan Klein, Tetsushi Oka, Richard Reeve

Division of Informatics, University of Edinburgh[*]*, Scotland, UK*

Abstract

Godot is a mobile robot platform that serves as a testbed for the interface between a sophisticated low-level robot navigation and a symbolic high-level spoken dialogue system. The interesting feature of this combined system is that information flows in two directions: (1) The navigation system supplies landmark information from the cognitive map used for the interpretation of the user's utterances in the dialogue system. (2) The semantic content of utterances analysed by the dialogue system are used to adjust probabilities about the robot's position in the navigation system.

1 Introduction

Many complete indoor mobile robot systems have been presented [3, 12, 16, 8]. Navigation is an essential task in the design of a mobile robotic agent. In our system, we want to fulfill all navigation functions by means of established [23, 18, 12, 11] and new techniques [21]. Furthermore, our system takes natural communication between the robot and human beings into account to help in navigation.

In particular, we present an integrated dialogue and navigation system on a mobile robot with the aim of investigating the use of a natural language such as English to help with navigation problems. Uncertainty is a major problem for navigation systems in mobile robots—interaction with humans in a natural way, using English rather than a programming language, would be a means of overcoming difficulties with localisation. Such situations—humans helping mobile robots to find their way or to complete tasks while engaging in a dialogue—are expected to become more widespread as robots begin to appear in domestic environments.

The dialogues we are considering are of the following nature: (1) the human informs the robot about its current position ("You are in my office."), possibly in response to a question from the robot after finding out that it is uncertain of its location ("Is this the kitchen?", "Is this the kitchen or Tim's office?"); (2) the human queries the robot about its current beliefs about its position ("Where are you?"), possibly followed by a correction or confirmation by the human; (3) the human instructs the robot to move to a certain position ("Go to the kitchen!"). Interactions described in (1) and (2) cause the robot to update its beliefs about its current position, whereas (3) would involve the robot planning the shortest path to that location and moving to it.

We set several requirements for the overall system. Communication should be in natural, unrestricted spoken English (as long as it is within the application domain), and in everyday usage of language. So rather than informing the robot that it is at "grid cell (4.2,8.9)", we would like the robot to understand natural descriptions such as "the kitchen", "a corridor" or more complicated expressions such as "my office", and "the corridor which leads to the emergency exit", including synonyms and phrases that do not necessarily have a unique designator. This not only has an impact on the design of the cognitive map, but also requires ontological knowledge and a semantic representation of the dialogue which enables the robot to perform inference. Further, utterance interpretation should be sensitive to the context. So, the semantic formalism we choose should be able to deal with anaphoric and deictic pronouns, presuppositions conveyed in utterances, and other referring expressions.

In the following sections we describe the most important parts of the combined navigation and dialogue handling system in more detail. We start by describing the low level navigation system components (Section 2). This includes a description of the cognitive map and how landmarks represented in it are linked to descriptions in semantic representations for utterances handled by the dialogue component. Section 3 contains the description of the spoken dialogue system component and presents the context-sensitive semantic representations used by the dialogue manager to represent the meaning of the dialogue, as well as a description of the implementation. We conclude in Section 4.

[*]This project is a joint effort between the Institute for Communicating and Collaborative Systems and the Mobile Robotics Group of the University of Edinburgh.

2 The Navigation System

In mobile robotics, *navigation* is a generic term for a spectrum of different functions which enable the robot to move autonomously in its environment. Although denied in the pure behaviour-based paradigm [2, 1], mobile robots often need some sort of internal representation of the environment, commonly referred to as a *cognitive map*. Our system uses three different layers of representation of the environment, a geometric, a topological and a semantic layer. The last layer is the connection point between the robot's knowledge about its environment (the map) and a symbolic natural language processing system, as presented in Section 3.

Based on the cognitive map, the robot can plan and execute motions. In addition to that, localisation errors need to be corrected and new sensor readings to be interpreted and integrated into the cognitive map. The algorithms dealing with each of these partial aspects need to run concurrently. Using our navigation software, Godot can travel through corridors and rooms of our department building. A complete description of the navigation system can be found in [21].

2.1 Godot the robot

Godot is a Real World Interfaces™ Magellan Pro mobile robot platform with an on-board control PC running Linux (Fig. 1).

Figure 1: *Talking to Godot*

Godot has a cylindrical shape, and is about 50 cm high and 41 cm in diameter. The sensor equipment consists of 16 sonar, infrared and collision sensors built in segments all around the circular base of the robot. Furthermore, it is equipped with two wheel encoders and a CCD camera mounted on a pan-tilt unit. It is driven by two independently controlled main wheels and has a support wheel in the back part. The on-board computer is connected to the local network via a wireless LAN interface.

2.2 Map building and administration

The navigator uses three layers of representation for the cognitive maps simultaneously, a geometric layer, a topological layer and a semantic layer.

On the first layer we use a probabilistic occupancy grid [5] to represent probabilities for occupied and free space on a regular subdivision of the environment. For each cell of the grid we save the probability of the cell being occupied space using the knowledge of past sensor readings r^t for this cell up to the current point T in time $P(occ_{xy} \mid r^{(1)}, r^{(2)}, \ldots, r^{(T)})$. We model the grid as a 0-dimensional Markov Random Field which allows us to treat the occupancy probabilities of the cells as independent random variables. For the interpretation and integration of sensor readings we use probabilistic sensor models (see Section 2.4).

The topological map is automatically constructed from the geometric map by subdividing the free space in the occupancy grid into distinct topological regions corresponding to rooms or parts of the corridor. As Fig. 2 illustrates, the borders between adjacent regions are found by searching for points on a Generalised Voronoi Diagram, a form of skeletonization, which minimises the clearance to the free space boundary [11, 23]. The topological map can be represented as an undirected graph in which regions are represented as nodes, and connections between them as edges.

Figure 2: *Screenshot of navigator interface*

The semantic layer extends the topological map by attaching symbolic labels to each region, hence regions become symbolic identifiers for a set of geometric map cells. The symbolic identifiers can be complex semantic structures, and are therefore able to represent formulations such as "Tim's office" or "the blue room" (see Section 3).

2.3 Motion planning

The navigation system can plan motions on all three layers of representation. Movements on the geomet-

ric layer are planned by means of a distance transform path planner, a relaxation algorithm [12]. The algorithm marks each free space cell with its minimum distance to the goal cell and performs a steepest descent search on this distance field to plan a path from a start cell to the goal cell [21]. Motion planning on the topological layer as well as the semantic layer reduces to a shortest path planning in the undirected graph of the topological map. Topological path plans are translated into geometric path plans by making the distance transform algorithm compute trajectories between centres of the topological regions that have to be traversed.

2.4 Sensor models and map building

Due to their effectiveness in free space detection, the main source of information for building geometric occupancy grids and for updating the geometric maps are the sonar sensors of Godot. For interpretation of the distance readings as occupancy probabilities we tested a learned as well as a manually designed probabilistic sensor model. The learned model is implemented as a multi-layer neural network which outputs occupancy probability values for occupancy grid cells given sonar data as input. We trained the network using a back-propagation training algorithm and a cross-entropy error function [15]. The training sets were built from real measurements with the sonar sensors at different positions in our building. Learning a sensor model has the big advantage that small peculiarities in the physical sensor characteristics are implicitly included into the model by the learning algorithm (see also [23]).

The manually designed model is not totally physically plausible but tailored to our purpose. With this model the sensors report very high probability values in an interval around the estimated obstacle, very low values in front and 0.5 behind to represent the lack of evidence (for details see [21]). Fig. 3 shows some examples of sonar scans which were interpreted with each of these models.

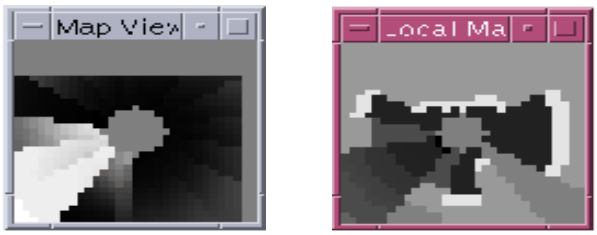

Figure 3: *Sonar scans with learned (l) and manually designed (r) sensor model (darker color means lower occupancy probability)*

Every time the robot administers three geometric occupancy grids concurrently. The global occupancy grid is the current representation that Godot has of its whole work space. It can be learned by joysticking the robot around or manually designed. In short time steps Godot performs sonar scans for a square area around itself containing current occupancy estimates. A sequence of several concurrent such estimates is merged into a local occupancy grid of the same size and position using Bayes's Rule [15]. The local occupancy grid is now used for position correction, and if desired, the local sonar scan can also be included into the global occupancy grid by means of Bayes's Rule to model changes in the environment.

For evaluation we also implemented an alternative *posemap* to implement belief map navigation based on algorithms by Thrun [22, 24]. Thrun's algorithm comprises two steps: sensing and action. The robot keeps track of a probability density function across a grid map of its environment, representing the likelihood of it being at a given *pose*, a position combined with an orientation, at a particular time. Each action taken by the robot shifts the probability density function and increases the spread (uncertainty), to account for errors introduced by moving. Each sensing provides the robot with landmark information which it can use to increase the certainty of its location. The posemap itself is represented as a three-dimensional array, where the x and y axes are equal to the corresponding axes of the cognitive map, and the z axis defines a particular orientation. Each cell in this array represents a unique robot pose.

2.5 Position correction

A main problem in mobile robotics is the reliable self-localisation of the robot under all circumstances. Especially on common office-floor carpet as used in our department building (see Fig. 1) errors accumulated in the wheel encoder readings become very quickly intolerable. Therefore, we implemented a 4-component position correction method [21].

The first component deals with a systematic error in the odometry by means of a linear correction of the robot's heading depending on the travelled distance. We thereby replace the internal position computation by our own one.

In addition we use *a priori* knowledge of the structure of the corridors in our department to find the current heading of the robot with respect to the walls. The infrared sensors of Godot give quite reliable distance readings to nearby obstacles even at a comparably high angular offset. Knowing that the walls are parallel to each other on every side of the corridor and that they always intersect at ninety degree angles, the navigation system uses the orientation of line segments fitted to at least four neighbouring IR sensor readings to find the heading of the robot [23].

For the correction of the position with respect to the global map, Godot continuously builds local smaller sized occupancy grids using the sonar sensor readings (see Section 2.4). These local grids are tested for a best match in correlation with the global map and the local grid in a local neighbourhood in pose space. The difference between the current internal position and the estimated best match location is used for correcting the internal position (see also [18]).

A fourth source of information for position correction is the *distance cross matching*. The comparison of four orthogonal IR readings and four distance measures along the major coordinates axes in the geometric map are used to compute correction terms for the internal (x,y)-position.

These four sources of information for position correction compute correction factors at each time step independently. These factors are weighted and summed up to the final corrections. By this means we try to combine several different self-localisation methods to increase the stability and applicability under different circumstances.

2.6 Software architecture

Godot's navigation system is implemented as a collection of communicating CORBA processes which extend the robot's basic Mobility™ control software and which run in parallel. Fig. 4 shows the different components, links between the boxes indicate possible interactions between the processes.

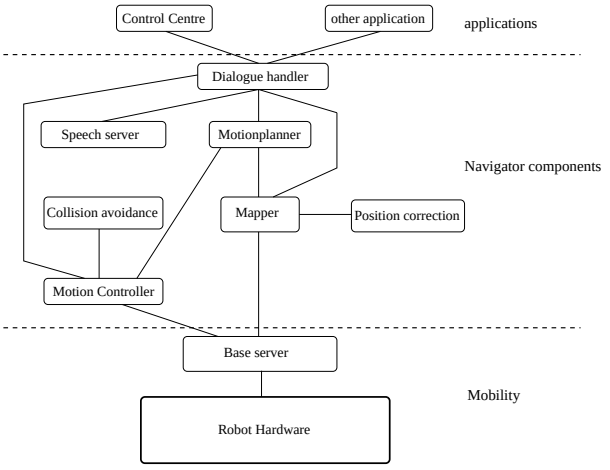

Figure 4: *Navigation system architecture*

The central component is the Mapper server which administers the maps on all layers and integrates new sensor readings. The path planner and the collision avoidance run concurrently, they can send commands to a motion controller which arbitrates between them according to priority rules. The motion planner interfaces with the base server delivered with the robot to set motor speeds. Applications can interface with the navigator by means of a dialogue handler which provides a simple command language for basic movements between topological regions or grid cells, and which allows to query the semantic map.

3 The Spoken Dialogue System

3.1 Representing and interpreting dialogue

The dialogue component uses Discourse Representation Structures (DRS) from Discourse Representation Theory (DRT) to represent the meaning of the dialogue between human and robot. DRT [9] is a well understood framework and covers a wide variety of linguistic phenomena including context-sensitive expressions such as pronouns and presuppositions. Further motivation for choosing DRT as semantic formalism is based on the fact that there are computational implementations available that provide means to extend existing linguistic grammars with DRS-construction and ambiguity resolution tools. An example DRS is shown in Fig. 5.

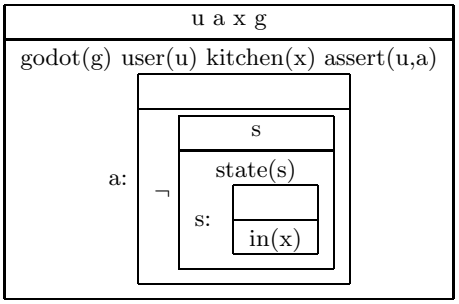

Figure 5: *Example DRS paraphrasing the utterance "You are not in the kitchen.".*

DRT gives us also the possibility to implement inference. There is a direct link between DRT and first-order logic because there is a translation from DRSs to formulas of first-order logic that behaves linearly in the size of the input. Inference helps to detect inconsistent information states, to rule out interpretations due to ambiguity resolution, and to exclude possible positions of the robot's cognitive map.

Regions of the topological map are labelled with DRSs. At the moment these are implemented as *a priori knowledge*, but future versions of the system envisage updating this information though interaction with users. For instance, the DRS for the utterance "You're in an office" might be assigned to a previously unknown region, and later be refined by "This is Tim's office". Dialogue updates integrate the DRS of the robot's current position (which might be a set of possible positions), and therefore make all inferences situation dependent. For instance, the user saying "You are in the kitchen" will rule out any region that is inconsistent with being in the kitchen.

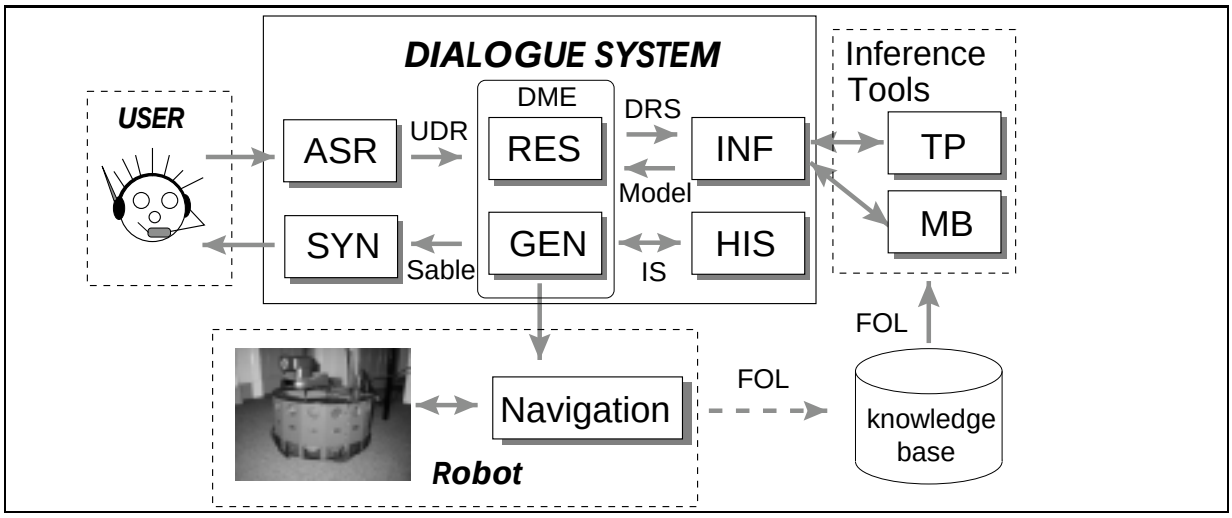

Figure 6: *Spoken-dialogue system architecture and interface to the navigation system in Godot.*

3.2 Implementation

The dialogue system is implemented as a collection of agents within version 2.1.0 of the Open Agent Architecture, OAA [4]. OAA is a piece of middleware supporting C++, Java, Lisp and Prolog, which enables one to put components together as a working dialogue system in a prototyping environment, where agents can run on different machines. Agents (roughly corresponding to the different components) communicate via *solvables*, specific queries that can be solved by certain agents. Although the agents are connected to one central facilitator, in the inference-based dialogue system there is a separate functional hierarchy governing them. Fig. 6 illustrates these dependencies, and shows what kind of messages are used as interfaces between them and in what order they are sent between components. We will now describe each of these agents in more detail.

The ASR agent (Automated Speech Recognition) is implemented as the off-the-shelf (speaker independent) Nuance 7.0 speech recogniser [17]. The Nuance recogniser requires an application specific speech recognition grammar. The grammar we use for Godot is compiled from a linguistic unification grammar and includes semantic representations. This means that the output of the ASR agent is actually a semantic representation, more specifically it is an *underspecified discourse representation* (UDR). UDRs are proto-DRSs with still unresolved information (anaphora and scope). The SYN agent, implemented as the Festival synthesiser [20], synthesises utterances coded in SABLE format (an XML standard for speech synthesis markup [19]).

The DME updates the information state with respect to new utterances (from the user) and decides the next move of the system. The information state (IS) stored in the dialogue history HIS is fairly straightforwardly structured, and describes a record consisting of a stack of 'last-moves' and an ordered list of pairs of DRSs and first-order models generated for these DRSs. Dialogue updates are carried out in a rule-based fashion, where the effects of update rules are applied to the information state if their pre-conditions hold. One of the update rules consults the resolution component RES to perform contextual resolution (resolving ambiguities arising from anaphoric expressions or scope bearing operators), mapping the UDR (stored in 'last-moves') into pairs of DRSs and models. Other update rules use the GEN component to generate utterances from the DRS, or generate actions from the model.

The INF agent is a mediator for inference facilities, using both theorem provers (the TP agent) and model builders (the MB agent) to find either models or counter-proofs. Incoming calls (DRSs) are distributed among TP and MB agents, and only the first result is returned to RES, whereas all inference jobs that are doomed to fail or not anymore required are killed. The TP agent takes the DRS, translates it into a first-order logic formula, consults the knowledge base for supporting background knowledge, and attempts to prove the negation of the resulting formula by giving it to the theorem prover SPASS [26]. The MB agent, on the other hand, tries to generate a model for the translated DRS and background knowledge, using the model builder MACE [14].

Finally, an OAA-CORBA bridge converts OAA-solvables into CORBA requests and forms the interface between the dialogue and navigation system. Implemented requests are returning the regions the robot believes it is currently in, returning all the regions known from its current map, returning the semantic representation labelling the region, and in-

structing Godot to move to a certain region.

4 Results and Future Work

What makes Godot an interesting robot is the interface between the low-level navigation system and the advanced spoken-dialogue component. There are several other robot prototypes that can be compared to Godot [7, 10, 13, 25]. Closest in functionality to Godot is probably the office robot Jijo-2 [6], which also has a navigation system but only makes use of a topological map and a Bayesian Network. Its spoken-dialogue system is similar in that it keeps track of the context, but unlike Godot, Jijo-2 doesn't have any generic reasoning capabilities.

Experiments with a first prototype of Godot (featuring an earlier, less advanced dialogue system) in a basement environment consisting of two corridors and an office, comprising five different topological regions (see Fig. 2), showed that Godot is capable of planning and executing collision free paths through narrow corridors in real-time and at a cruising speed of 12 cm/s. The position correction works robustly in narrow corridors, but reveals problems in larger areas without detectable walls. Queries of the cognitive map and move commands requiring path planning can be issued to the robot and processed in real-time and at any time while only using the on-board computer for computation.

We are currently defining test environments to evaluate Godot and measure performances with the advanced version of the dialogue system and alternative configurations of the navigation system. Future work will also involve changing or enhancing semantic descriptions of the topological map via dialogue.

References

[1] R. Brooks. A robust layered control system for a mobile robot. *IEEE journal of Robotics and Automation*, 2(1):14–23, 1986.

[2] R. Brooks. Intelligence without representation. *Artificial Intelligence*, 47:139–159, 1991.

[3] J. Buhmann and W. Burgard. The mobile robot RHINO. *AI Magazine*, 16:1, 1995.

[4] A. Cheyer and D. Martin. The open agent architecture. *Journal of Autonomous Agents and Multi-Agent Systems*, 4(1/2):143–148, March 2001.

[5] A. Elfes. *Occupancy Grids: A Probabilistic Framework for Robot Perception and Navigation*. PhD thesis, Department of Electrical and Computer Engineering, Carnegie Mellon University, 1989.

[6] J. Fry, H. Asoh, and T. Matsui. Natural dialogue with the jijo-2 office robot. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems IROS-98*, pages 1278–1283, 1998.

[7] A. Green and K. Severinson-Eklundh. Task-oriented dialogue for cero: a user-centered approach. In *Proceedings of Ro-Man'01 (10th IEEE International Workshop on Robot and Human Communication)*, pages 146–151, Bordeaux-Paris, 2001.

[8] I. Horswill. Polly: A vision-based artificial agent. In *Proceedings of the 11th National Conference on Artificial Intelligence*, pages 824–829, Menlo Park, CA, USA, 1993. AAAI Press.

[9] H. Kamp and U. Reyle. *From Discourse to Logic; An Introduction to Modeltheoretic Semantics of Natural Language, Formal Logic and DRT*. Kluwer, Dordrecht, 1993.

[10] T. Längle, T. C. Lüth, E. Stopp, G. Herzog, and G. Kamstrup. KANTRA – A Natural Language Interface for Intelligent Robots. In *Intelligent Autonomous Systems (IAS 4)*, pages 357–364, 1995.

[11] J. C. Latombe. *Robot Motion Planning*. Kluwer Academic Publishers, 1991.

[12] D. Lee. *The Map-Building and Exploration Strategies of a Simple Sonar-Eqipped Mobile Robot*. PhD thesis, Cambridge University, 1996.

[13] O. Lemon, A. Bracy, A. Gruenstein, and S. Peters. A Multi-Modal Dialogue System for Human-Robot Conversation. In *Proceedings of NAACL 2001*, 2001.

[14] W. McCune. Automatic Proofs and Counterexamples for Some Ortholattice Identities. *Information Processing Letters*, 65:285–291, 1998.

[15] T. M. Mitchell. *Machine Learning*. Computer Science Series. McGraw Hill International Editions, 1997.

[16] N. J. Nilsson. Shakey the robot. Technical Report 323, SRI International, 1984.

[17] Nuance Communications, Inc., 1005 Hamilton Avenue, Menlo Park, CA 94025 USA. *Nuance Speech Recognition System Version 7.0 Nuance Application Developer's Guide*, 2000.

[18] A. C. Schultz and W. Adams. Continuous localization using evidence grids. In *IEEE proceedings Robotics and Automation*, pages 2833–2839, 1998.

[19] R. Sproat, A. Hunt, M. Ostendorf, P. Taylor, A. Black, and K. Lenzo. Sable: A standard for tts markup. In *ICSLP98*, pages 1719–1724, 1998.

[20] P. A. Taylor, A. Black, and R. Caley. The architecture of the festival speech synthesis system. In *The Third ESCA Workshop in Speech Synthesis*, pages 147–151, 1998.

[21] C. Theobalt. Navigation on a mobile robot. Master's thesis, University of Edinburgh, 2000.

[22] S. Thrun. Bayesian landmark learning for mobile robot localization. *Machine Learning*, 33(1):41–76, 1998.

[23] S. Thrun. Learning maps for indoor mobile robots. *Artificial Intelligence*, 99(1):21–71, 1998.

[24] S. Thrun, W. Burgard, and D. Fox. A probabilistic approach to concurrent mapping and localization for mobile robots. *Machine Learning*, 31(1-3):29–53, 1998.

[25] M. C. Torrance. Natural communication with robots. Master's thesis, Department of Electrical Engineering and Computer Science, Cambridge MA, 1994.

[26] C. Weidenbach, B. Afshordel, U. Brahm, C. Cohrs, T. Engel, E. Keen, C. Theobalt, and D. Topic. System description: Spass version 1.0.0. In H. Ganzinger, editor, *16th International Conference on Automated Deduction, CADE-16*, volume 1632 of *LNAI*, pages 314–318. Springer, 1999.