Available online at www.sciencedirect.com
ScienceDirect
Computer Speech and Language 24 (2010) 395–429
COMPUTER SPEECH AND LANGUAGE
www.elsevier.com/locate/csl

Evaluation of a hierarchical reinforcement learning spoken dialogue system

Heriberto Cuayáhuitl *, Steve Renals, Oliver Lemon, Hiroshi Shimodaira

Institute for Communicating and Collaborative Systems, School of Informatics, University of Edinburgh, 10 Crichton Street, Edinburgh EH8 9AB, Scotland, UK

Received 5 June 2008; received in revised form 15 June 2009; accepted 3 July 2009
Available online 14 July 2009

Abstract

We describe an evaluation of spoken dialogue strategies designed using hierarchical reinforcement learning agents. The dialogue strategies were learnt in a simulated environment and tested in a laboratory setting with 32 users. These dialogues were used to evaluate three types of machine dialogue behaviour: hand-coded, fully-learnt and semi-learnt. These experiments also served to evaluate the realism of simulated dialogues using two proposed metrics contrasted with 'Precision-Recall'. The learnt dialogue behaviours used the Semi-Markov Decision Process (SMDP) model, and we report the first evaluation of this model in a realistic conversational environment. Experimental results in the travel planning domain provide evidence to support the following claims: (a) hierarchical semi-learnt dialogue agents are a better alternative (with higher overall performance) than deterministic or fully-learnt behaviour; (b) spoken dialogue strategies learnt with highly coherent user behaviour and conservative recognition error rates (keyword error rate of 20%) can outperform a reasonable hand-coded strategy; and (c) hierarchical reinforcement learning dialogue agents are feasible and promising for the (semi) automatic design of optimized dialogue behaviours in larger-scale systems.

Keywords: Spoken dialogue systems; Hierarchical reinforcement learning; Human–machine dialogue simulation; Dialogue strategies; System evaluation

1. Introduction

A spoken dialogue system can be defined as consisting of four interlinked modules: speech understanding, dialogue management, response generation, and a knowledge base. In a human–machine dialogue, a user's spoken utterance is received as a speech waveform, which may have been distorted, by the speech understanding module which extracts a user dialogue act from the speech signal. The dialogue act is entered into the machine's knowledge base, and the machine then updates its dialogue state using information extracted from the knowledge base. The machine dialogue state is used by the dialogue manager to choose a machine dialogue

* Corresponding author. Tel.: +44 52 3315731360.
E-mail address: h.cuayahuitl@ed.ac.uk (H. Cuayáhuitl).

0885-2308/$ - see front matter © 2009 Elsevier Ltd. All rights reserved.
doi:10.1016/j.csl.2009.07.001

act which is then used by the response generation module to produce a corresponding machine speech signal in reply to the user. This is a cyclical process, illustrated in Fig. 1, which continues until one of the participants in the conversation (human or machine) ends the dialogue.

In this paper we are primarily concerned with the dialogue manager. Given the current state of the dialogue, the principal role of the dialogue manager is to choose an action, which will result in a change of dialogue state. The strategy followed by the dialogue manager, sometimes referred to as the *policy*, should be designed to enable successful, efficient and natural conversations. This is a challenging goal, and in most fielded spoken dialogue systems the dialogue manager is handcrafted by a human designer. This hand-crafted approach is limited since it is not always easy to specify the optimal action at each state of the dialogue, a dialogue behaviour for the entire user population which is generic and static is usually assumed, and designing such strategies is labour-intensive, especially for large systems.

Since the mid-1990s a number of researchers have explored the development of automatic algorithms that can specify a dialogue strategy. In particular, reinforcement learning approaches (Sutton and Barto, 1998) have been used to optimize a machine's dialogue behaviour (Levin and Pieraccini, 1997; Walker et al., 1998; Levin et al., 2000; Young, 2000). In this scenario, a conversation is regarded as a sequence of dialogue states, with the machine receiving a reward for executing an action inducing a state transition in the conversational environment, as illustrated in Fig. 2.

A reinforcement learning dialogue agent aims to learn its behaviour from interaction with an environment, where situations are mapped to actions by maximizing a long-term reward signal. Briefly, the standard reinforcement learning paradigm works by using the formalism of Markov Decision Processes (MDPs) (Kaelbling et al., 1996; Sutton and Barto, 1998; Russell and Norvig, 2003). An MDP is characterized by a set of states S, a set of actions A, a state transition function, and a reward or performance function that rewards the agent for each selected action. Solving the MDP means finding a mapping from the current state s_t to an action a_t corresponding to a dialogue policy $\pi^*(s_t)$:

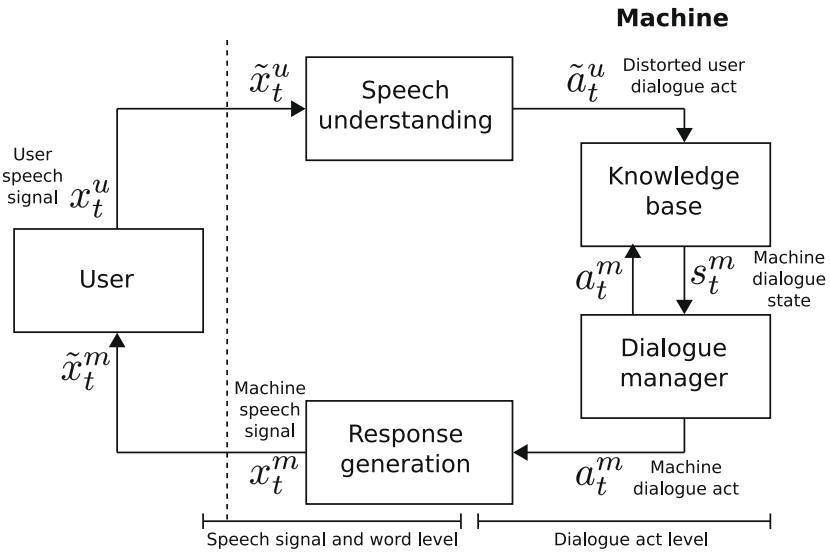

Fig. 1. A pipeline architecture of speech-based human–machine communication, where dialogue state s_t^m is used by the dialogue manager to choose action a_t^m. Modelling the dialogue strategy at the semantic level allows us to omit the speech signal and word levels.

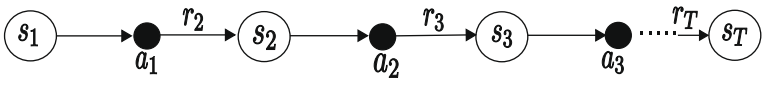

Fig. 2. A dialogue of length T described in terms of a state sequence s_t, with state transitions induced by actions a_t.

$$\pi^*(s_t) = \arg\max_{a_t \in A} Q^*(s_t, a_t). \quad (1)$$

The Q function specifies the cumulative rewards for each state–action pair. An alternative but more computationally intensive model for sequential decision-making under uncertainty is the partially observed MDP (POMDP). In a POMDP the dialogue state is not known with certainty (as opposed to an MDP), and solving it means finding a mapping from belief states to actions (Roy et al., 2000; Williams and Young, 2007).

Most previous work on dialogue strategy learning has aimed at obtaining a single global solution (Levin et al., 2000; Walker, 2000; Young, 2000; Singh et al., 2002; Scheffler and Young, 2002; Pietquin, 2004; Williams, 2006; Young et al., 2007). The optimization of dialogue strategies has been carried out using two main approaches: corpus-based approaches (Walker, 2000; Litman et al., 2000) which make use of an experimentally collected set of dialogues for training the dialogue strategy (or some aspects of it); and simulation-based approaches (Scheffler and Young, 2002; Frampton, 2008; Rieser, 2008; Henderson et al., 2008) in which a simulation environment including a user model is employed to generate simulated dialogues for training.

A dialogue strategy may not require complete world knowledge, nor is it always necessary for the whole action set to be available at each state. In this paper we address such issues using a *hierarchical sequential decision making* approach, in which dialogue states can be described at different levels of granularity, and an action can execute behaviour using either a single dialogue act or a composite sub-dialogue. This approach offers several benefits. First, modularity helps to solve sub-problems that may be easier to solve than the whole problem. Second, sub-problems may include only relevant dialogue knowledge in the states and relevant actions, thus reducing significantly the size of possible solutions: consequently they can be found faster. Finally, there is the possibility to reuse sub-solutions when dealing with new problems. These properties are crucial for learning the behaviour of large-scale spoken dialogue systems in which there may be a large set of state variables or a large number of actions. The cost of this approach is that optimal solutions may not be guaranteed; however, this suboptimality may be well worth the gains in terms of scalability to large systems.

This paper has two main contributions. First, we have developed and evaluated a heuristic simulation environment used to learn dialogue strategies in an automatic way. Second, we have developed and evaluated hierarchical spoken dialogue behaviours learnt using a Semi-Markov Decision Process (SMDP) to address the problem of scalable dialogue optimization, described in more detail in Cuayáhuitl (2009). We have compared these hierarchical, or 'semi-learnt' behaviours, with both hand-crafted and fully-learnt behaviours and we have found that the semi-learnt behaviours are more suited to deployment. Our evaluations have been carried out in tests with real users in the context of a spoken dialogue system in the travel planning domain.

The rest of the paper is organized as follows: Section 2 describes the dialogue simulation environment. Section 3 briefly describes the hierarchical reinforcement learning dialogue agents. Section 4 describes the travel planning spoken dialogue system. Section 5 reports an evaluation of machine dialogue behaviours. Section 6 provides a quantitative evaluation of the simulated environment. Finally, sections 7 and 8 discuss and summarize our findings.

2. A heuristic dialogue simulation environment

A simulation environment for human–machine conversations involves modelling the dynamics of everything that is outside the dialogue manager. This section presents a heuristic simulation approach for generating human–machine conversations based on dialogue acts. The proposed approach generates both coherent and distorted conversations, useful for testing and learning dialogue strategies for information-seeking mixed-initiative spoken dialogue systems. This approach does not require data for training the models in the simulation environment (this is useful in scenarios where dialogue data does not exist), as it uses heuristics to simulate the dynamics of task-oriented conversations based on dialogue acts. It employs two main simulation models—simulated user behaviour and ASR error simulation—which are shown at the bottom of Fig. 3. The first simulation model (on the right of the figure) generates coherent user responses, i.e. responses that make sense to humans. Here it was assumed that real users behave in a coherent fashion, based on user dialogue acts that are consistent according to a user Knowledge Base (KB) that keeps the history of the conversation. This is a strong assumption and its validity is addressed later. The second model distorts coherent user dialogue acts due to imperfect speech recognition and understanding. The distorted user responses and data-

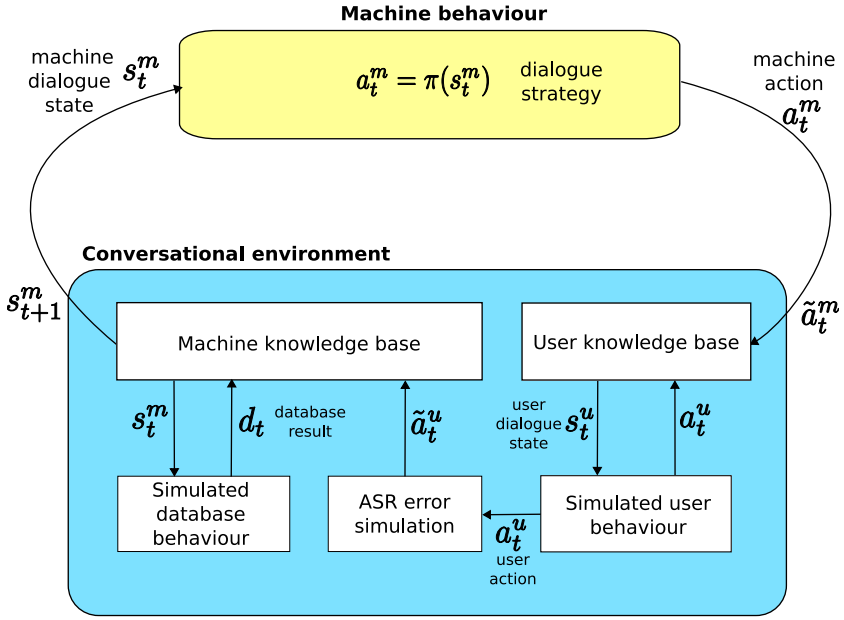

Fig. 3. The agent–environment interaction for simulating human–machine conversations, useful for learning or testing dialogue strategies for spoken dialogue systems.

base results update the machine's KB so that the dialogue strategy can choose actions accordingly. The proposed dialogue simulator uses an ontology to represent the conversant's knowledge base.

Fig. 3 shows the agent–environment interaction for human–machine dialogue simulation. The interaction is as follows: the machine is in a given dialogue state s_t^m, and emits dialogue act a_t^m by following dialogue strategy $\pi(s_t^m)$. A distorted machine dialogue act $\tilde{a}_t^m$ (machine response[1]) is fed into the user's KB to observe the user dialogue state s_t^u, from which an action a_t^u is taken (user response). This user response is distorted with ASR errors into $\tilde{a}_t^u$, and is fed into the machine's KB. The machine action may require interaction with simulated database behaviour by sending queries and retrieving database results d_t. Then the next machine state s_{t+1}^m is observed from the machine's current KB. Once the machine is in a new state, it takes another dialogue act, and so on until the end of the conversation.

2.1. Modelling conversational behaviour

A human–machine dialogue can be modelled by the perceptions and actions of both conversants. Fig. 4 shows the dynamics of communication at the dialogue act level. The conversants use two sources of knowledge at different levels of granularity: *knowledge-rich states* k_t (also referred to as "knowledge base") to represent all possible perceptions about the conversation, and *knowledge-compact* states s_t to represent a compact version of the current dialogue state. The latter are used for action selection.

Algorithm 1 specifies the high-level steps for simulating a task-oriented human–machine dialogue. Briefly, the algorithm starts initializing parameters for the knowledge bases (ontologies of dialogue entities) of both conversants. The algorithm invokes three simulated behaviours: the machine's dialogue strategy π_t^m, the user's dialogue strategy π_t^u, and the distorter of machine/user dialogue acts δ. A conversant at a time interacts with its partner as follows: (a) observes the current knowledge-compact state, (b) selects an appropriate dialogue act type, (c) generates a dialogue act with the current dialogue act type in context, (d) distorts such dialogue act to simulate misrecognitions or misunderstandings, (e) updates its knowledge-rich state with the undistorted dialogue act, and (f) updates the knowledge-rich state of its partner with the distorted dialogue act. This process iterates until one of the conversants terminates the dialogue.

[1] The reason for distorting machine responses was to model user confusions.

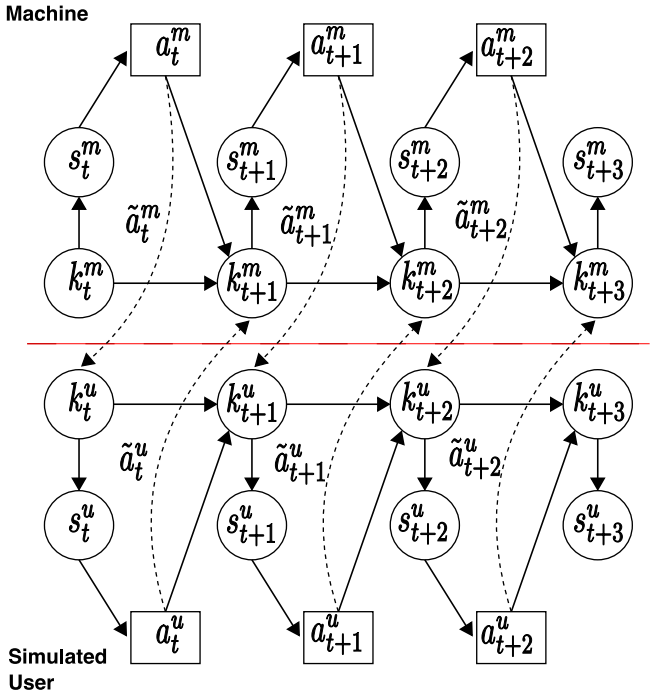

Fig. 4. Dynamics of human–machine communication at the dialogue act level (this diagram does not follow the conventions of dynamic Bayesian networks). A conversant in a knowledge-rich state k_t, observes a knowledge-compact state s_t, and takes dialogue act a_t in order to feed it to its knowledge-rich state and convey it to its partner, received distortedly as $\tilde{a}_t$. The current knowledge k_t, action a_t and partner response determine the next knowledge-rich state k_{t+1}, and so on until the end of the dialogue.

Algorithm 1: Simulator of task-oriented human–machine conversations

function HUMANMACHINEDIALOGUESIMULATOR( )
  $k_0^m \leftarrow$ initialize machine knowledge-rich state
  $k_0^u \leftarrow$ initialize user knowledge-rich state
  $t \leftarrow$ initialize time-step to 0
  repeat
    $s_t^m \leftarrow$ observe machine dialogue state from k_t^m
    $a_t^m \leftarrow$ choose machine dialogue act type following $\pi^m(s_t^m)$
    Generate machine dialogue act $\equiv$ dialogue act type a_t^m in context
    $\tilde{a}_t^m \leftarrow$ get distorted dialogue act from $\delta(a_t^m, k_t^m)$
    Update k_t^m with a_t^m and update k_t^u with $\tilde{a}_t^m$
    $s_t^u \leftarrow$ observe user dialogue state from k_t^u
    $a_t^u \leftarrow$ choose user dialogue act type following $\pi^u(s_t^u)$
    Generate user dialogue act $\equiv$ dialogue act type a_t^u in context
    $\tilde{a}_t^u \leftarrow$ get distorted dialogue act from $\delta(a_t^u, k_t^u)$
    Update k_t^u with a^u and update k^m with $\tilde{a}_t^u$
    $t \leftarrow t + 1$
  until one of the conversants terminates the conversation
end function

Enumerating all possible machine or user dialogue acts usually results in large sets. Therefore, our approach assumes that action selection of both conversants is based on *dialogue act types* rather than dialogue acts. This is beneficial because for task-oriented conversations a small set of dialogue act types can be

employed. Table 1 shows the core dialogue act types that define the behaviour of our human–machine simulated conversations. The user dialogue act types are a subset of the ones used by Georgila et al. (2005), and the set of machine dialogue act types are an extension of the ones used by Walker and Passonneau (2001). Based on this, the agent selects dialogue act types following dialogue strategy π^m, and the user selects dialogue act types following dialogue strategy π^u. Once an action has been chosen, it takes context into account so that conversations can be generated at the dialogue act level. Context is given by the dialogue state, which specifies the slot in focus, slots to fill or confirm, etc. A sample machine action for requesting the slot date is "$a_t^m = req(date)$" and a corresponding sample user response is "$a_t^u = pro(date = 01dec2007, time = morning)$".

Based on this, the user takes actions following dialogue strategy π^u defined by:

$$\pi^u(s_t^u) = \begin{cases} pro, & \text{if last machine action } a^m \text{ is a request or offer,} \\ con, & \text{if last action } a^m \text{ is a correct explicit confirmation or incorrect} \\ & \text{explicit confirmation (the latter with some probability, e.g. 0.2),} \\ rep, & \text{if last action } a^m \text{ is an apology or incorrect confirmation,} \\ sil, & \text{otherwise} \end{cases} \quad (2)$$

and the machine takes actions a following dialogue strategy $\pi^m(s_t^m)$ defined by:

$$a = \begin{cases} ope, & \text{if first time step,} \\ req, & \text{if unknown slot in focus,} \\ sic + req, & \text{if unknown slot in focus and Single Slot to Confirm (SSC),} \\ mic + req, & \text{if unknown slot in focus and Multiple Slots to Confirm (MSC),} \\ apo + req, & \text{if slot in focus with low confidence level,} \\ sec, & \text{if slot in focus with medium confidence level and SSC,} \\ mec, & \text{if slot in focus with medium confidence level and MSC,} \\ acc, & \text{if slot in focus with high confidence level,} \\ dbq + sta, & \text{if null database result and confirmed non-terminal slots,} \\ pre + ofr, & \text{if database result with few uninformed tuples,} \\ apo + ofr, & \text{if terminal slot with low confidence level,} \\ ofr, & \text{if unconfirmed terminal slot and db tuples presented before,} \\ ack, & \text{if unacknowledged dialogue goal and confirmed terminal slot,} \\ rel, & \text{if empty database result and confirmed non-terminal slots,} \\ clo, & \text{otherwise.} \end{cases} \quad (3)$$

Although the strategy π^u may not include all possible realistic behaviours, it yields coherent behaviour, and its evaluation is addressed later. Finally, the hand-crafted strategy π^m acts as the baseline strategy in our experiments using reinforcement learning agents as described in the next sections.

2.2. Speech recognition error simulation

Due to the fact that current Automatic Speech Recognition (ASR) technology is far from perfect, errors have to be modelled in the simulated environment. This simulation model operated with a two-stage approach. First, slot values of user dialogue acts were distorted with probability $p(user) = 0.2$ in order to model machine confusions; applying equal amounts of insertions, substitutions and deletions. Second, slot values were assigned with random confidence levels; they were assigned the well known three-tiered confidence levels (low, medium, high) to indicate their speech recognition confidence. In addition, slot values of machine dialogue acts were distorted with probability $p(agent) = 0.1$ in order to model user confusions.

Table 1
Dialogue Act Types (DATs) for task-oriented human–machine spoken dialogues. Abbreviations: IC = implicit confirmation and EC = explicit confirmation.

Agent	ID	DAT	Sample utterance
User	*pro*	provide	I want a flight from Edinburgh to London
	rep	reprovide	I said a flight to London from Edinburgh
	con	confirm	Yes, please
	sil	silence	[*remain in silence*]
Machine	*req*	request	And, what is your destination city?
	apo	apology	I am sorry, I didn't understand that
	sic	single_IC	A flight to London
	mic	multiple_IC	A flight from Edinburgh to London
	sec	single_EC	I think you said London, is that correct?
	mec	multiple_EC	I heard from Paris to London, is that right?
	acc	accept_slot	[*move to next ascending slot with lowest value*]
	dbq	db_query	[*performs a database query*]
	ofr	offer	Would you like A, B or C?
	sta	status	Please wait while I query the database
	pre	present	The cost of this flight is 120 pounds
	rel	relax	Try again with some different information
	ack	acknowledgement	All right, this flight has been booked
	ope	opening	Welcome to the travel planning system
	clo	closing	Thank you for calling, good bye

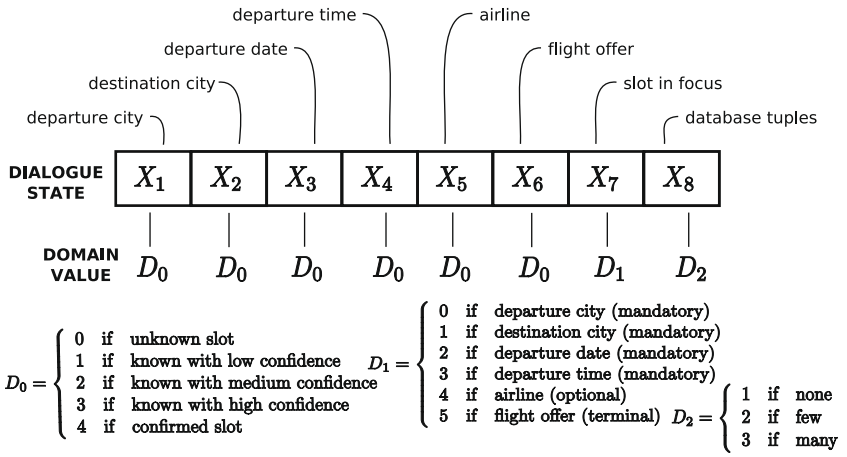

Fig. 5. Dialogue state for the flight booking spoken dialogue strategy. Each variable X_i with domain values D_0 has five possible values, variable X_7 has six possible values, and variable X_8 has 3 possible values, resulting in $5^6 \times 6 \times 3 = 281250$ states.

2.3. An illustrative decision-making problem

Consider that you have the task of designing a spoken dialogue strategy for a flight booking system. In such a system the user can say things such as '*a flight from London to Prague for the twenty second of October in the morning travelling with KLM*'—alternatively, the user may provide the information across several shorter utterances. A dialogue strategy is a mapping from dialogue states to actions and specifies the system's way of behaving. The dialogue state (used to describe the current situation in the conversation) is defined by a vector of state variables as illustrated in Fig. 5. This decision-making problem has 281,250 states $\left(\prod_{X_i \in X} |X_i|\right)$. A sample dialogue using this state space is shown in Table 2. Table 3 shows the same sample dialogue but including dialogue acts generated as described in the previous two sub-sections. The mapping from states to actions can be done either manually, or using flat reinforcement learning methods, or using hierarchical reinforcement

Table 2
Sample human–machine dialogue in the flight booking domain, where the dialogue state is formed by the state variables shown in Fig. 5, and a set of actions is available per state. At this point action-selection is arbitrary (before learning).

Agent	Dialogue state	Action	Sample utterance
SYS	0,0,0,0,0,0,0,1	req	Tell me your flight information
USR			*A flight from Edinburgh to Paris*
SYS	3,2,0,0,0,0,0,1	acc	
	3,2,0,0,0,0,2,1	mic+req	OK, a flight from Edinburgh to Rome, when do you want to travel?
USR			*I said 'to Paris'*
SYS	4,3,0,0,0,0,2,1	sic+req	OK, a flight to Paris, when do you want to fly?
USR			*December first, flying with British Airways*
SYS	4,4,1,0,2,0,2,1	apo+req	I'm sorry, when do you want to travel?
USR			*On the first of December*
SYS	4,4,3,0,2,0,2,1	acc	
	4,4,3,0,2,0,3,1	mic+req	Travelling on the first of December with British Airways, at what time do you prefer your flight?
USR			*In the morning*
SYS	4,4,4,1,4,0,3,1	apo+req	Sorry, at what time do you prefer to fly?
USR			*In the morning*
SYS	4,4,4,3,4,0,3,1	sec	Did you say in the morning?
USR			*Yes please*
SYS	4,4,4,4,4,0,5,1	dbq+sta	Please wait while I retrieve information...
USR			
SYS	4,4,4,4,4,0,5,2	pre+ofr	All right, flights from Edinburgh to Paris on the first of December...[present information] Would you like choice 1, choice 2 or choice 3?
USR			*The second one please*
SYS	4,4,4,4,4,2,5,2	sec	Did you say choice two?
USR			*That's right*
SYS	4,4,4,4,4,4,5,2		[terminal state]

Notes: This example is used for illustration purposes and shows basic behaviour of more complex interactions. Abbreviations: req = request, acc = accept *w*/high confidence, mic = multiple implicit confirmation, sic = single implicit confirmation, apo = apology, sec = single explicit confirmation, dbq = database query, sta = status of dialogue, pre = present information, and ofr = offer choices.

learning methods as described in the rest of this paper. The benefits of reinforcement learning include automatic design and optimal behaviour according to a performance measure, while hierarchical methods aim to have a more practical application for systems with large state–action spaces.

3. Hierarchical reinforcement learning dialogue agents

A critical problem in flat Reinforcement Learning (RL) is scalability since it operates with a single policy that behaves by executing only primitive actions. The size of state spaces grows exponentially with the number of state variables incorporated into the environment state—the 'curse of dimensionality'. As a result, RL agents find solutions only very slowly. *Temporal abstraction* addresses these problems by incorporating hierarchical structures into RL agents. This is attractive for dialogue systems for several reasons. First, human decision-making activity occurs in sequential courses of action, where decisions do not happen at each step, but rather in temporally extended activities following their own policies until termination (Barto and Mahadevan, 2003). Second, hierarchical decision makers can solve more complex problems than flat ones (Dietterich, 2000). Third, task-oriented dialogues have shown evidence of following hierarchical structures (Grosz and Sidner, 1986; Litman and Allen, 1987). This section presents two forms of hierarchical dialogue control and a case study that is evaluated in the rest of the paper.

Table 3
Sample dialogue in the flight booking system. Although simulations are only based on dialogue acts, an equivalent wording is given for a better understanding. This dialogue shows a sample speech recognition error after the first user utterance.

Agent	Dialogue act	Wording
SYS	req(dep_city)	Tell me your flight information
USR	pro(dep_city=edinburgh, des_city=paris)	*I would like a flight from Edinburgh to Paris*
ASR		*[I would like a flight from Edinburgh to Rome]*
SYS	mic(dep_city=edinburgh, des_city=rome)+req(date) OK, a flight from Edinburgh to Rome, when do you want to travel?	
USR	rep(des_city=paris)	*I said 'to Paris'*
SYS	sic(des_city=paris)+req(date)	OK, a flight to Paris, when do you want to fly?
USR	pro(date=01dec, airline=british_airways) *December first, flying with British Airways*	
SYS	apo(date)+req(date)	I'm sorry, when do you want to travel?
USR	pro(date=01dec)	*On the first of December*
SYS	mic(date=01dec, airline=british_airways)+req(time)	Travelling on the first of December with British Airways, at what time do you prefer your flight?
USR	pro(time=morning)	*In the morning*
SYS	apo(time)+req(time)	Sorry, at what time do you prefer your flight?
USR	rep(time=morning)	*In the morning*
SYS	sec(time=morning)	Did you say in the morning?
USR	con(time=yes)	*Yes please*
SYS	dbq(db_query)+sta(db)	Please wait while I retrieve information...
USR		
SYS	pre(db_result)+ofr(flight)	All right, flights from Edinburgh to Paris on the first of December in the morning travelling with British Airways: [present information] Would you like choice 1, choice 2 or choice 3?
USR	pro(flight=choice2)	*The second one please*
SYS	sec(flight=choice2)	Did you say choice two?
USR	con(flight=yes)	*That's right*

Note: more complex information presentation is beyond the scope of this work.

3.1. Dialogue control using Semi-Markov Decision Processes

We formulate spoken dialogue control as a discrete Semi-Markov Decision Process (SMDP), following Dietterich (2000). A discrete-time SMDP $M = \langle S, A, T, R \rangle$ is characterized by a set of states S; a set of actions A; a transition function T that specifies the next state s' given the current state s and action a with probability $P(s', \tau | s, a)$; and a reward function $R(s', \tau | s, a)$ that specifies the reward given to the agent for choosing action a when the environment makes a transition from state s to state s'. The random variable τ denotes the number of time-steps taken to execute action a in state s. The SMDP model allows temporal abstraction, where actions take a variable amount of time to complete their execution. In this model two types of actions can be distinguished: (a) single-step actions roughly corresponding to dialogue acts and (b) multi-step actions corresponding to sub-dialogues. Fig. 6 illustrates a conceptual dialogue at runtime with states s_t, actions a_t and rewards r_t. Whilst the full dialogue and child dialogue execute primitive and composite actions, the grandchildren dialogues execute only primitive actions. Note that the execution of primitive actions yields single rewards and the execution of composite actions lasting τ time steps yields cumulative discounted rewards given at time $t + \tau$.

In this paper, we treat each composite dialogue action as a separate SMDP as described in Cuayáhuitl et al. (2007) and Cuayáhuitl (2009). In this way an MDP can be decomposed into multiple SMDPs hierarchically organized into L levels and N models per level, denoted as $\mathscr{M} = \left\{M_j^i\right\}$, where $j \in \{0, \ldots, N-1\}$

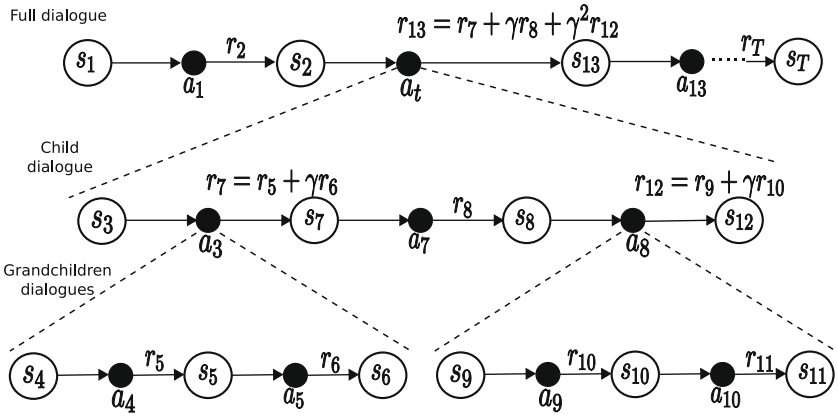

Fig. 6. Conceptual hierarchical dialogue at runtime with states s_t, actions a_t (lasting τ time steps) and rewards $r_{t+\tau}$. Actions a_t can be either primitive or composite, the former yield single rewards and the latter yield cumulative discounted rewards.

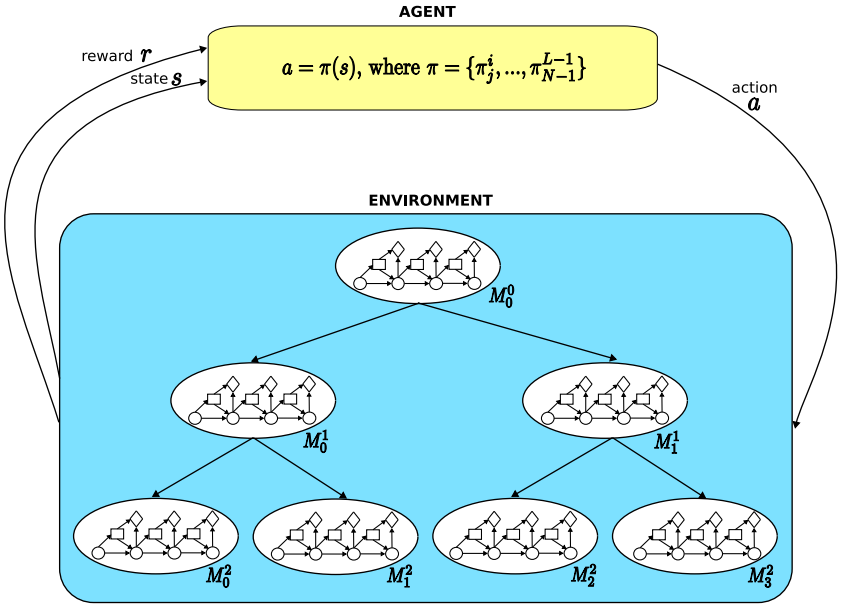

Fig. 7. Architecture of the agent–environment interaction using multiple SMDPs M^i_j, circles represent states, squares represent actions, and diamonds represent rewards.

and $i \in \{0, \ldots, L-1\}$. Thus, any given SMDP in the hierarchy is denoted as $M^i_j = \left\langle S^i_j, A^i_j, T^i_j, R^i_j \right\rangle$, see the environment of Fig. 7 for an illustration.

The goal in an SMDP is to find an optimal policy π^*, that maximizes the reward of each visited state. The optimal action-value function $Q^*(s,a)$ specifies the expected cumulative reward for executing action a in s and then following π^*. The Bellman equation for Q^* of subtask M^i_j can be expressed as

$$Q^{*i}_j(s,a) = \sum_{s',\tau} P^i_j(s',\tau|s,a)\left[R^i_j(s',\tau|s,a) + \gamma^\tau \max_{a'} Q^{*i}_j(s',a')\right], \tag{4}$$

where the discount rate $0 \leqslant \gamma \leqslant 1$ makes future rewards less valuable than immediate rewards as it approaches 0. Finally, the optimal policy for each subtask is defined by

$$\pi^{*i}_j(s) = \arg\max_{a \in A^i_j} Q^{*i}_j(s,a). \tag{5}$$

These policies can be found by dynamic programming or reinforcement learning algorithms for SMDPs. For instance, the HSMQ-Learning algorithm of Dietterich (2000) approximates Eq. (4) according to

$$Q_j^i(s,a) \leftarrow (1-\alpha)Q_j^i(s,a) + \left[r_\tau + \gamma^\tau \max_{a'} Q_j^i(s',a')\right]. \quad (6)$$

This behaviour (also referred to as *fully-learnt*) receives rewards in the following form when executing actions a lasting τ time steps:

$$r = r_1 + \gamma r_2 + \gamma^2 r_3 + \cdots + \gamma^{\tau-1} r_\tau. \quad (7)$$

The HSMQ-Learning algorithm converges to optimal context-independent policies (Dietterich, 2000). Although this is a weaker form of optimality than other forms for being only locally optimal, context-independent policies facilitate state abstraction (useful to compress the state) and policy reuse.

3.2. Dialogue control using constrained hierarchical SMDPs

The behaviour of reinforcement learning dialogue agents can be constrained with prior expert knowledge, aiming to combine behaviours specified by human designers and inferred automatically (Paek and Pieraccini, 2008). In this direction we have reported an approach using reinforcement learning with Hierarchical Abstract Machines (HAMs) (Cuayáhuitl et al., 2006). HAMs are used to reduce the available actions per state, similar to non-deterministic finite state machines whose transitions may invoke lower-level machines, each machine specifying a sub-dialogue. Because the HAMs approach does not overcome the curse of dimensionality, we extend the previous form of dialogue control by constraining each hierarchical SMDP with some prior expert knowledge. For such a purpose, we associate a HAM denoted as H_j^i to SMDP M_j^i in order to specify the prior knowledge (Cuayáhuitl, 2009). In this way, dialogue control can be seen as executing two decision-making models in parallel: a HAM, and a hierarchy of SMDPs. Each HAM partially specifies the behaviour of its corresponding subtask, and therefore constrains the actions that a reinforcement learning agent can take in each state. For such a purpose, a cross product of models per subtask is used, referred to as *induced SMDP* $M_j'^i = H_j^i \circ M_j^i$. Briefly, the cross product operates as follows: (1) the induced state space uses joint states $(s,\bar{s})$, where s is an *environment state* in SMDP M_j^i and $\bar{s}$ is a *choice state* in HAM H_j^i; (2) a HAM tells its corresponding SMDP the available actions at state s; (3) the transition functions of both models are executed in parallel; and (4) the SMDP's reward function rewards each chosen primitive action. In this joint model the HAMs make decisions in states with a single action, and the policies of the SMDPs make decisions in states with multiple actions.

This form of behaviour (also referred to as *semi-learnt*) is based on the SMDP state s and the HAM choice state $\bar{s}$. Using a more compact notation for the joint dialogue state $w = (s,\bar{s})$ (Marthi et al., 2006), the Bellman equation for the action-value function of induced subtask $M_j'^i$ can be expressed as

$$Q_j^{*i}(w,a) = \sum_{w',\tau} P_j^i(w',\tau|w,a)\left[R_j^i(w',\tau|w,a) + \gamma^\tau \max_{a'} Q_j^{*i}(w',a')\right]. \quad (8)$$

Optimal context-independent policies for the Q-value function above can be found by combining the algorithms HAMQ-Learning (Parr and Russell, 1997) and HSMQ-Learning (Dietterich, 2000) using the following update rule, see (Cuayáhuitl, 2009) for more details:

$$Q_j'^i(w_t,a_t) \leftarrow (1-\alpha)Q_j'^i(w_t,a_t) + \left[r_{t+\tau} + \gamma^\tau \max_{a'} Q_j'^i(w_{t+\tau},a')\right]. \quad (9)$$

3.3. Case study: a travel planning dialogue system

This is a multi-goal mixed-initiative spoken dialogue system in the travel planning domain, allowing users to book single flights, return flights, hotels and cars. It supports the following features: hand-crafted or learnt dialogue strategies, multiple goals within a single dialogue, and implicit switching across flight dialogue goals.

For *fully-learnt behaviour*, the state space is described in Table 4 using a hierarchy of 21 dialogue subtasks. This hierarchy employed 43 non-binary state variables, 15 primitive actions and 20 composite actions. The latter correspond to the child subtasks. The reward function focused on efficient conversations (i.e. the shorter the dialogue the better), and is defined by the following rewards given to the agent for choosing action a when the environment makes a transition from state s to state s':

$$r(s,a,s') = \begin{cases} 0, & \text{for successful (sub)dialogue,} \\ -10, & \text{for an already collected subtask } M_j^i, \\ -10, & \text{for collecting subtask } M_i^i \text{ before } M_{i-1}^i, \\ -10, & \text{for presenting many/none items of information,} \\ -10, & \text{for multiple greetings or closings,} \\ -10, & \text{for executing action } a \text{ and remaining in state } s' = s, \\ -1, & \text{otherwise.} \end{cases} \quad (10)$$

For *semi-learnt behaviour*, the state–action space was similar to that of fully-learnt behaviour. The difference here is that the dialogue subtasks M_j^i were extended with Hierarchical Abstract Machines (HAMs) H_l^k, where their cross product yields the induced subtasks $M_j^{ri} = H_l^k \circ M_j^i$. The hierarchy of induced subtasks for the travel planning system is shown in Fig. 8, and used the abstract machines described in Figs. 9 and 10 (prohibiting apologies in medium and high confidence levels). These HAMs control the machine's dialogue behaviour in deterministic state transitions, but in stochastic state transitions the reinforcement learning agents optimized decision-making.

The learning parameters used by the algorithms were the same for both learning approaches. The learning rate parameter α decays from 1 to 0 according to $\alpha = 100/(100+\tau)$, where τ represents elapsed time-steps in

Table 4
State variables and actions of the subtask hierarchy in the travel planning system. Whilst a flat approach involves a large state–action space in the order of 10^{23} (Cuayáhuitl, 2009), our hierarchical representation is only using a space of $800K$ state–actions.

Subtask	State variables	Actions (*composite actions are* M_j^i)
M_0^0	GIF, SAL, G00, G03, G04, G05	M_0^1, M_2^2, M_3^2, M_4^2, gre, clo
M_0^1	GIF, G01, G02	M_0^2, M_1^2
M_0^2	DBT, END, MAN, OPT, TER	M_0^3, M_1^3, M_2^3, dbq+sta, rel
M_1^2	DBT, END, MAN, TER	M_3^3, M_4^3, dbq+sta, rel
M_2^2	DBT, END, INI, MAN, TER	M_5^3, M_6^3, M_7^3, dbq+sta, rel
M_3^2	DBT, END, INI, MAN, OPT, TER	M_8^3, M_9^3, M_A^3, M_B^3, dbq+sta, rel
M_4^2	DBT, END, MAN, TER	M_C^3, M_D^3, dbq+sta, rel
M_0^3	SIF, C00, C01, C02, C03, C04, C05	req, apo+req, sic+req, mic+req, sec, mec, acc
M_1^3	C6	req, apo+req, sec
M_2^3	ACK, END, PRE, C07	apo+ofr, sec, pre+ofr, ofr, ack
M_3^3	SIF, C15, C16	req, apo+req, sic+req, mic+req, sec, mec, acc
M_4^3	ACK, END, PRE, C17	apo+ofr, sec, pre+ofr, ofr, ack
M_5^3	C18	req, apo+req, sec
M_6^3	SIF, C19, C20, C21	req, apo+req, sic+req, mic+req, sec, mec, acc
M_7^3	ACK, END, PRE, C22	apo+ofr, sec, pre+ofr, ofr, ack
M_8^3	C23	req, apo+req, sec
M_9^3	SIF, C24, C25, C26, C27, C28	req, apo+req, sic+req, mic+req, sec, mec, acc
M_A^3	C29	req, apo+req, sec
M_B^3	ACK, END, PRE, C30	apo+ofr, sec, pre+ofr, ofr, ack
M_C^3	C31	req, apo+req, sec
M_D^3	ACK, END, PRE, C32	apo+ofr, sec, pre+ofr, ofr, ack

Values of state variables: Goal In Focus (GIF) ← {0 = flight booking, 1 = outbound flight, 2 = return flight, 3 = hotel booking, 4 = car rental, 5 = summarize trip}; Salutation (SAL) ← {0 = null, 1 = greeting, 2 = closing}; {G00, G01, G02, G03, G04, G05, INI, MAN, OPT, TER} ← {0 = unfilled subtask, 1 = filled subtasks, 2 = confirmed subtask}; Database Tuples (DBT) ← {0 = none, 1 = empty, 2 = few, 3 = many}; Slot In Focus (SIF) ← {Cij}; acknowledgement or current dialogue goal (ACK) ← {0,1}; Cij ← {0 = unfilled, 1 = low confidence, 2 = medium confidence, 3 = medium confidence, 4 = confirmed}; Status of information presentation of current dialogue goal (PRE) ← {0,1}; END ← {0 = continue, 1 = terminate current subtask}.

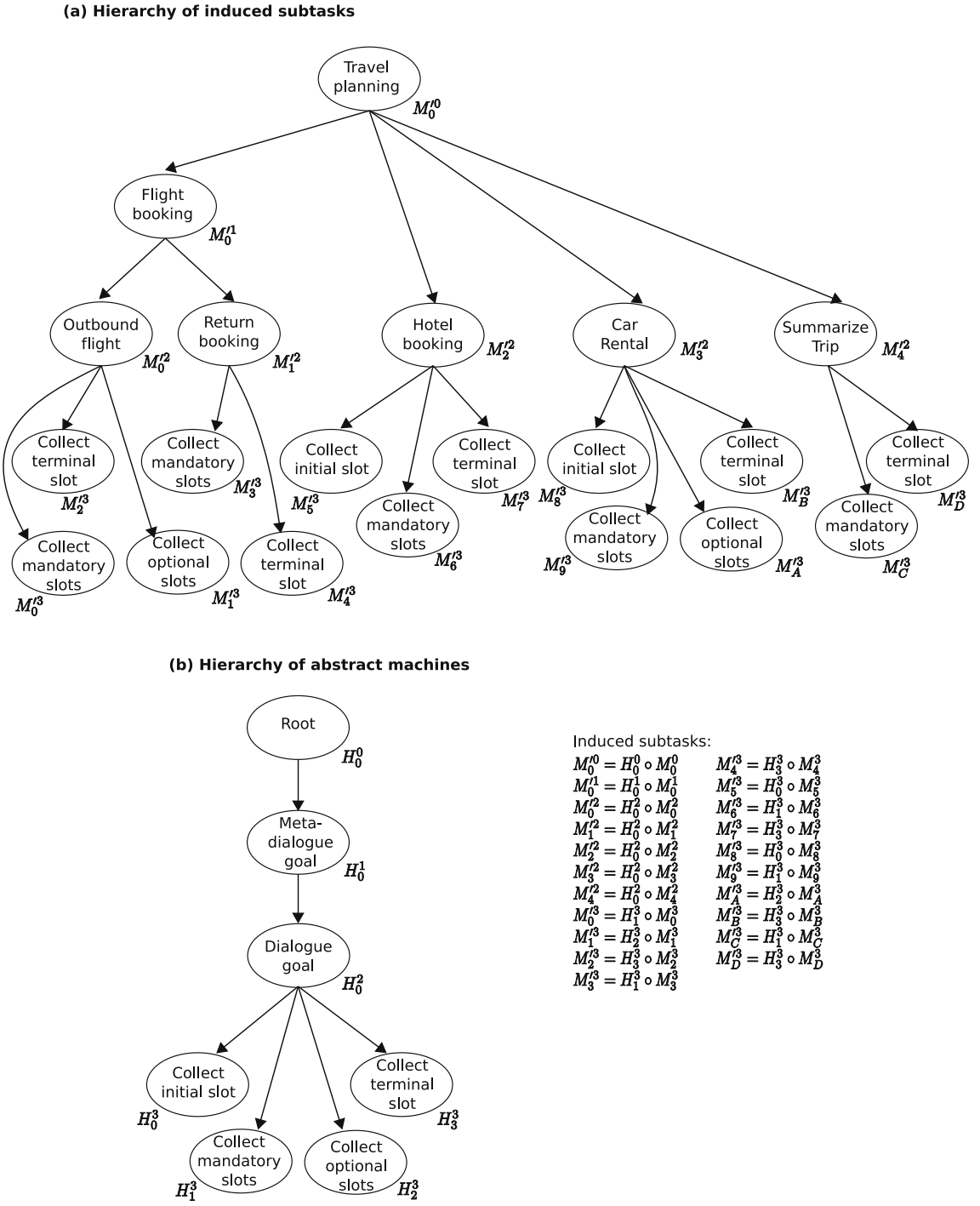

Fig. 8. A hierarchy of induced subtasks for the travel planning system. The abstract machines H_i^k are specified in Figs. 9 and 10, and the state variables for each dialogue subtask M_j^i are specified in Table 4.

the current subtask. Each subtask M_j^i had its own learning rate. The discount factor $\gamma = 1$ makes future rewards equally as valuable as immediate rewards, as in Singh et al. (2002). The action selection strategy used ϵ-Greedy with $\epsilon = 0.01$, and initial Q-values of 0. This choice of parameters satisfies the requirements for convergence to optimal context-independent policies.

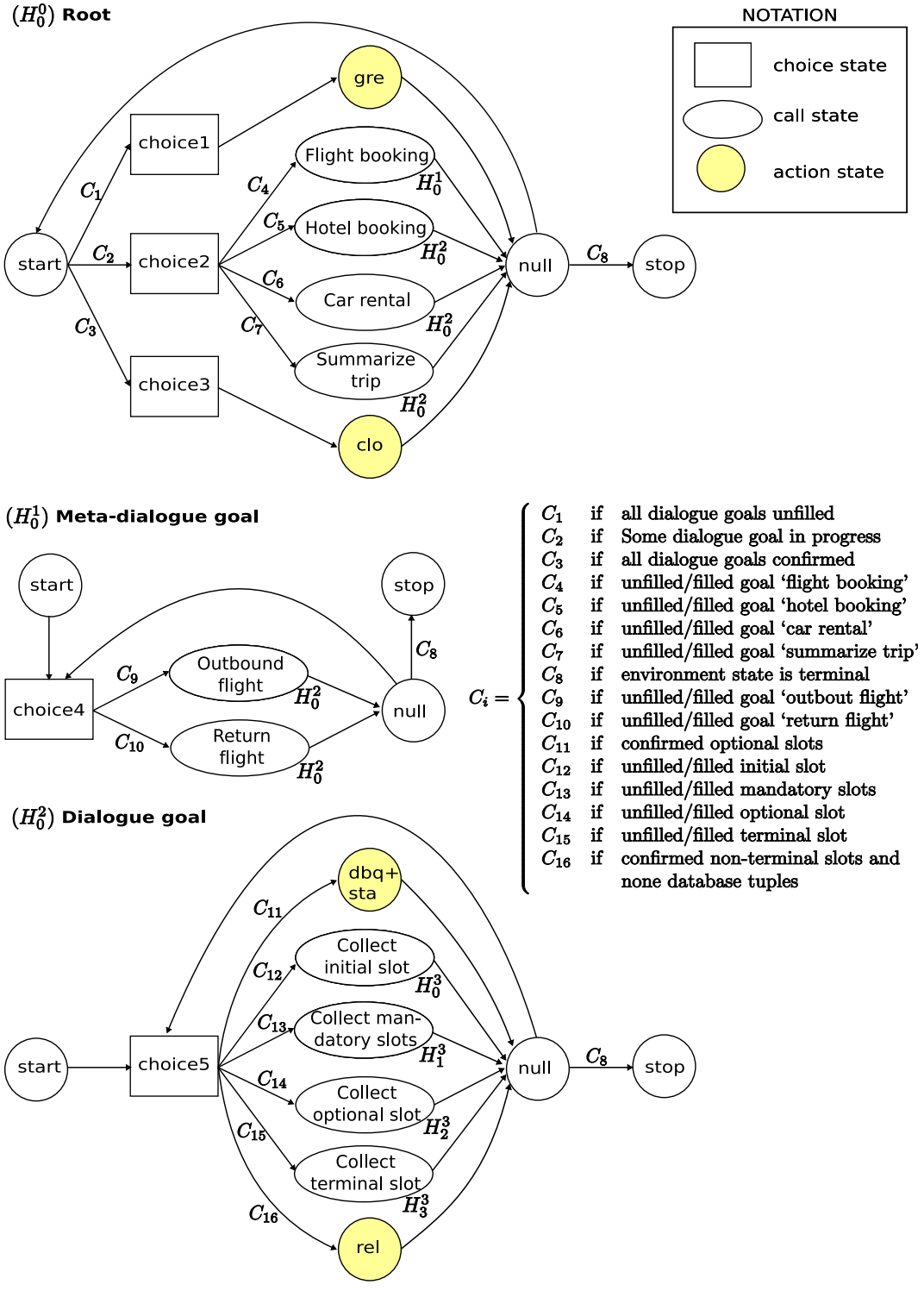

Fig. 9. Abstract machines for the travel planning spoken dialogue system (Part 1).

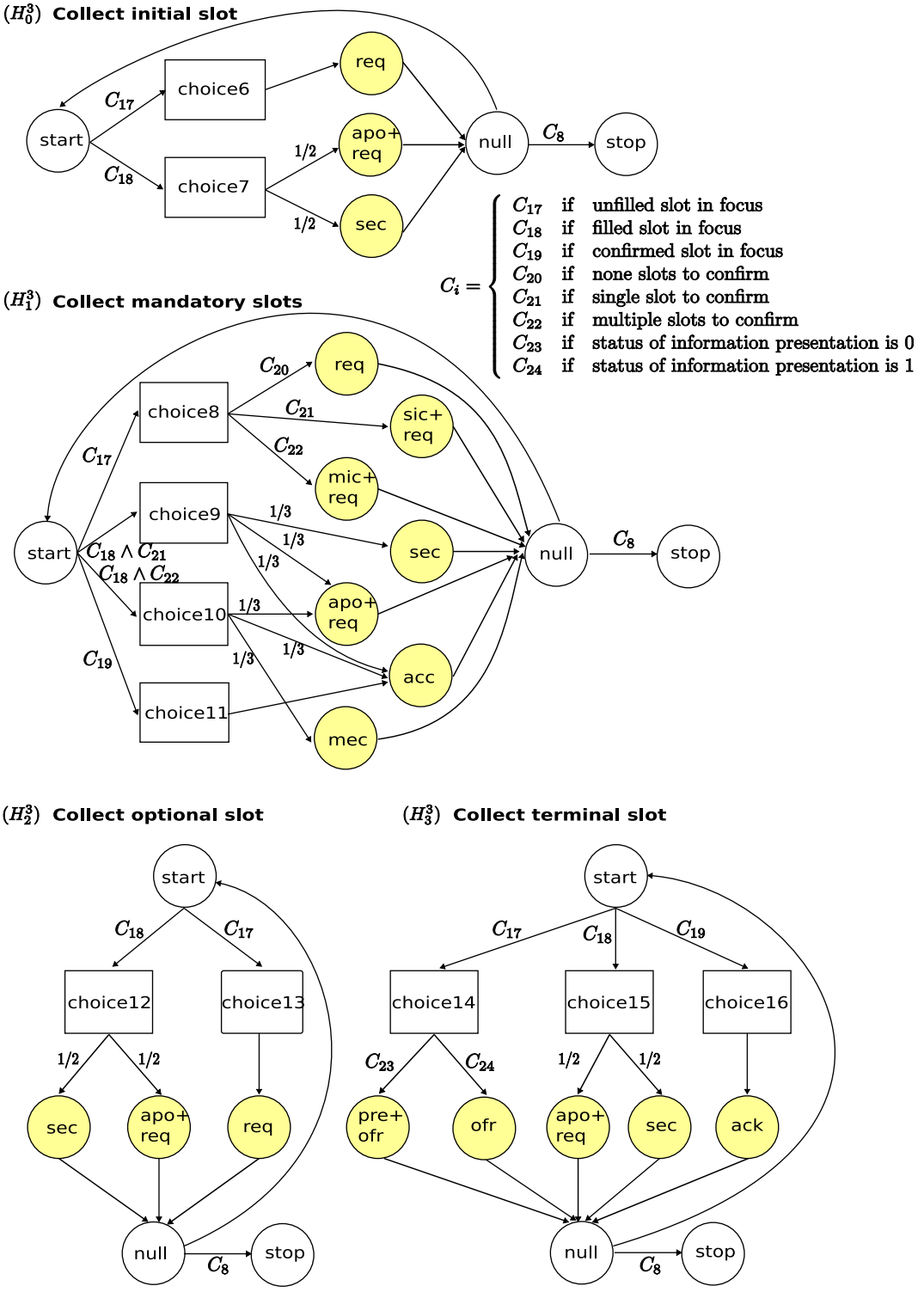

Fig. 10. Abstract machines for the travel planning spoken dialogue system (Part 2).

3.4. Qualitative description of hand-crafted and learnt dialogue strategies

The *hand-crafted* baseline strategy operated as follows: (a) if slot in focus is unknown then request information, with implicit confirmation if there were any filled slots; (b) if slot in focus is known with low confidence then do an apology; (c) if slot in focus is known with medium confidence then do an explicit confirmation; and (d) if slot in focus is known with high confidence then move the slot in focus to the next ascending one with lower value (this is also referred to as 'slot acceptance'). This behaviour is specified more concretely in Eq. (3), and its evaluation is reported in Section 6.3.

The *fully-learnt* behaviour was inferred by the approach described in Section 3.1. This strategy used all actions in every state and differs from the hand-crafted one by allowing acceptance, confirmation and rejection in any filled slot regardless of confidence level. For instance, action 'mic' can be chosen with low/medium/high confidence levels. Briefly, the learnt behaviour differed from the hand-crafted one in the use of more acceptances (action 'acc'), more multiple implicit confirmations (action 'mic'), fewer apologies (actions 'apo+req' and 'apo+ofr'), and fewer multiple explicit confirmations (action 'mec'). In addition, although the fully-learnt policy inferred the sequence of sub-dialogues, it used the same sequence as the other behaviours. Thus, the dialogue strategies of this paper differ in the selection of primitive (low level) actions rather than composite (high-level) actions.[2]

The *semi-learnt* behaviour was inferred by the approach described in Section 3.2. Similarly to the fully-learnt behaviour, this strategy differs from the hand-crafted one by allowing acceptance, confirmation and rejection in any filled slot regardless of confidence level. The semi-learnt behaviour differed from the hand-crafted one in the use of more acceptances (action 'acc'), more multiple implicit confirmations (action 'mic'), fewer apologies (actions 'apo+req' and 'apo+ofr'), and fewer multiple explicit confirmations (action 'mec'). However, it differs from the fully-learnt behaviour by constraining the actions available per state as shown in Figs. 9 and 10, where the semi-learnt policy prohibited apologies in slots with medium or high confidence levels. See Cuayáhuitl (2009) for a detailed quantitative comparison of dialogue strategies using simulated conversations.

4. Spoken dialogue system architecture

Our experiments were based on a travel planning spoken dialogue system supporting hand-crafted or learnt dialogue behaviour. The latter uses dialogue strategies designed by hierarchical reinforcement learning agents on a simulated environment. This system is based on the Open Agent Architecture (OAA) (Cheyer and Martin, 2001). Alternatively, other architectures can be used such as Galaxy-II (Seneff et al., 1998). Fig. 11 shows a high-level architecture using eight OAA-based agents in order to support speech-based task-oriented human–machine communication. The communication flows between facilitator (parent) and the other agents (children). Briefly, the user gives speech signals x_t^u corresponding to words w_t^u, concepts or slots c_t^u, and dialogue acts a_t^u. However, the machine understands them with distortions $(\tilde{w}_t^u, \tilde{c}_t^u, \tilde{a}_t^u)$, and answers back to the user with speech signals x_t^m corresponding to words w_t^m, slots c_t^m, and dialogue acts a_t^m. The user may also misunderstand the machine, and so on until one of the conversants terminates the conversation. The rest of this section describes each agent in the system.

4.1. Facilitator agent

OAA is an agent-based framework to build autonomous, flexible, fault-tolerant, distributed and reusable software systems (Cheyer and Martin, 2001). OAA agents can be written in multiple programming languages and run on a computer network with different operating systems. They have a parent *facilitator* agent, which coordinates the communication of child agents by keeping a knowledge base of their services. Child agents are service providers and service requesters. Service providers let the facilitator know of their own capabilities, and

[2] The benefit of learning the sequence of sub-dialogues is relevant for adaptive behaviour at different levels of granularity, and further experimentation using different sequences of sub-dialogues is left as future work.

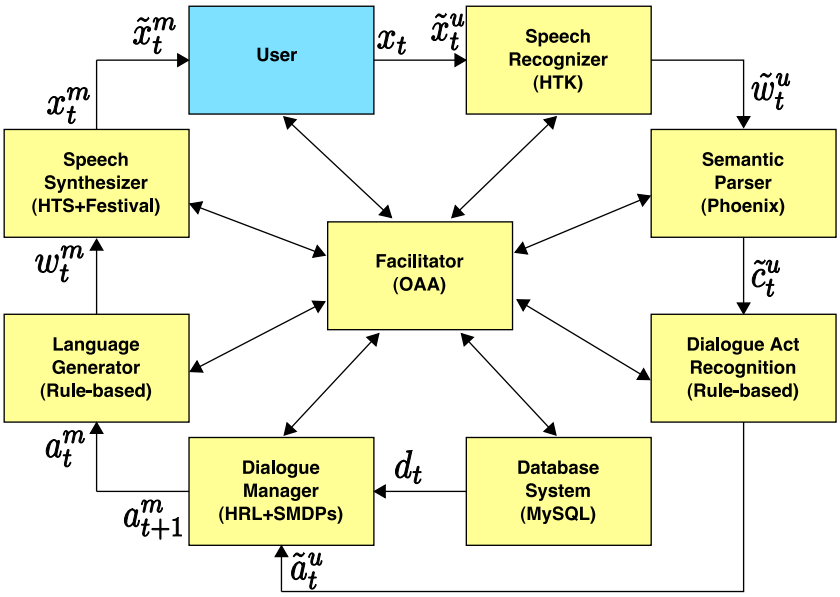

Fig. 11. Architecture of the CSTR travel planning spoken dialogue system supporting deterministic or learnt dialogue behaviour. Human–machine communication is carried out with speech signals x_t, words w_t, concepts or slots c_t, and dialogue acts a_t.

service requesters request capabilities from other agents. They communicate by passing string messages between child agents and facilitator.

4.2. Speech recognition agent

The task of the speech recognition agent was to receive user speech signals after each machine prompt w_t^m and to generate a word sequence including confidence levels $\tilde{w}_t^u$, derived from the recognition hypothesis incorporating confidence scores $\bar{w}_t^u$. This agent used the multithreaded ATK API, which is a layer on top of the HTK speech recognition libraries (Young, 2006, 2007). This agent used the acoustic models (trained with data from British speakers) generated from the TALK project,[3] and customized-based language models with a lexicon of 263 words. The confidence levels were assigned by dividing the confidence score range $[0 \ldots 1]$ into three equal areas, equivalent to l = low, m = medium, and h = high confidence. The following table illustrates this process.

ID	Event	Outcome
w_t^m	Machine prompt	Welcome to the CSTR travel planning system Tell me your flight information
w_t^u	User response	I would like a single flight from Edinburgh to Paris
$\bar{w}_t^u$	ASR hypothesis with confidence scores	how(0.27) about(0.31) a(0.15) single(0.60) flight(0.56) with(0.32) b._m._i.(0.47) from(0.70) edinburgh(0.59) to(0.40) paris(0.56)
$\tilde{w}_t^u$	ASR hypothesis w/conf. levels	how(l) about(l) a(l) single(m) flight(m) with(l) b._m._i.(m) from(h) edinburgh(m) to(m) paris(m)
w_{t+1}^m	Machine prompt	A single flight from Edinburgh to Paris travelling with BMI. When do you want to travel?
w_{t+1}^u	User response	I would like to travel with Air France...

[3] Our speech recognition and speech synthesis OAA agents used wrappers generated from the TALK project (Lemon et al., 2006).

4.3. Semantic parsing agent

The semantic parsing agent generated concept or keyword sequences $\tilde{c}_t^u$ from a (distortedly) recognized word sequence $\bar{w}_t^u$. This agent used the Phoenix spontaneous speech parser (Ward, 1994) that maps a word string into a semantic frame. A semantic frame is a set of slots of information, each slot with an associated context-free grammar. Such grammars are compiled into recursive transition networks, which are matched with the given word sequence by a top-down chart parsing algorithm. This agent used three frames (corresponding to flights, hotels and cars) including 18 semantic networks. See the table below for a sample parsed word sequence.

ID	Event	Outcome
w_t^m	Machine prompt	Welcome to the CSTR travel planning system Tell me your flight information
w_t^u	User response	I would like a single flight from Edinburgh to Paris
$\bar{w}_t^u$	ASR hypothesis wo/conf. scores	how about a single flight with b._m._i. from Edinburgh to Paris
$\tilde{c}_t^u$	Semantic parse	Flight:[FlightType].SINGLE Flight:[DepCity].[City].EDINBURGH Flight:[DesCity].[City].PARIS Flight:[Airline].BMI
$\tilde{a}_t^u$	User dialogue act	pro(FlightType=single.m,DepCity=edinburgh.m, DesCity=paris.m,Airline=bmi.m)
w_{t+1}^m	Machine prompt	A single flight from Edinburgh to Paris travelling with BMI. When do you want to travel?
w_{t+1}^u	User response	I would like to travel with Air France...

4.4. Dialogue act recognition agent

This agent generated user dialogue acts $\tilde{a}_t^u$ using a two-stage approach. First, a user dialogue act type was selected taking into account the current concept sequence $\tilde{c}_t^u$ and last machine dialogue act corresponding to the machine prompt w_t^m. Once a dialogue act type had been selected, it took context into account to become a user dialogue act $\tilde{a}_t^u$. Although it is possible to generate more than one dialogue act per user utterance, this agent generated a single user dialogue act (see the table above for an example).

4.5. Database system agent

This agent returned database tuples based on SQL queries from the dialogue manager. It used a web scraper to populate a local database, retrieving travel data from a commercial web site (http://www.opodo.co.uk). This strategy was selected for avoiding long time responses from direct queries to the web.

4.6. Dialogue management agent

The dialogue management agent is the key component to evaluate. It generated machine dialogue acts a_t^m from the hierarchy of policies π_j^i based on three different types of dialogue behaviours: deterministic (described in Section 2), fully-learnt and semi-learnt (described in Section 3, more details in Cuayáhuitl (2009)). Since these dialogue behaviours only differ in their action-selection mechanism, and the rest of the OAA-based agents (see Fig. 11) did not change regardless of the behaviour of choice, it is fair to say that these behaviours were evaluated under similar conditions.

4.7. Language generation agent

The task of the language generation agent was to generate a machine prompt w_t^m in natural language based on a template-based approach. A prompt template has a word sequence embedding variables, and was selected given the current machine dialogue act a_t^m, dialogue state s_t^m or joint state w_t^m, and a simple help mechanism.[4] Once a prompt template had been selected, it took context into account by replacing variables with values in the machine's knowledge base in order to generate the word sequence w_{t+1}^m. This agent included 463 prompt templates. The next table shows a sample prompt template c_t^m and its corresponding machine prompt w_{t+1}^m.

ID	Event	Outcome
w_t^m	Machine prompt	Welcome to the CSTR travel planning system Tell me your flight information
$\tilde{a}_t^u$	User dialogue Act	pro(FlightType=single.m,DepCity=edinburgh.m, DesCity=paris.m,Airline=bmi.m)
a_t^m	Machine Dialogue act	mic(FlightType=single,DepCity=edinburgh, DesCity=paris,Airline=bmi)+req(DepDate)
c_t^m	Prompt for action 'mic' Prompt for action 'req'	A $FlightType flight from $DepCity to $DesCity travelling with $Airline When do you want to travel?
w_{t+1}^m	Machine prompt	A single flight from Edinburgh to Paris travelling with BMI. When do you want to travel?
w_{t+1}^u	User response	I would like to travel with Air France...

4.8. Speech synthesis agent

The speech synthesis agent generated speech signals x_t^m from a given word sequence w_t^m. This agent is based on the Festival text-to-speech system[5] with an HTS voice generated from eight hours of recorded speech (Yamagishi et al., 2007). The speech signals were generated online, using a pre-processing stage to split word sequences at punctuation symbols in order to avoid long silences in the machine's utterance.

5. Spoken dialogue system evaluation

The aim of our experiments was to investigate if the learnt dialogue agents can outperform deterministic behaviour in a realistic environment. For such a purpose the spoken dialogue system described in the previous section was implemented and tested with a set of users, in laboratory conditions. Table 7 shows a sample dialogue.

5.1. Evaluation methodology

The CSTR travel planning spoken dialogue system was evaluated using a number of metrics, mostly derived from the PARADISE framework (Walker et al., 2000), which has been widely accepted for evaluating the performance of spoken dialogue systems.

(i) *Dialogue efficiency*: This group of quantitative metrics includes *system turns*, *user turns*, and *elapsed time* (in seconds). All of them report averages per dialogue goal (a conversation may have several dialogue goals). Elapsed time includes the time used by both conversants.

[4] Simple automatic help: (a) first slot collection = no help, (b) second collection = help prompt suggesting to fill multiple slots, (c) third collection: help prompt suggesting a shorter sentence, (d) fourth collection = help prompt suggesting to fill a single slot, and (e) others = help prompt suggesting to rephrase the sentence.

[5] <http://www.cstr.ed.ac.uk/projects/festival>.

(ii) *Dialogue quality*: These metrics consists of *Word Error Rate* (WER), *Keyword Error Rate* (KER), and *Event Error Rate* (EvER). The latter metric is useful because dialogue systems have to handle trade-offs among acceptance, confirmation and rejection events. The EvER metric is decomposed into the following metrics reported as percentages: *correct acceptances*, *correct confirmations*, *correct rejections*, *false acceptances*, *false confirmations* and *false rejections*. Other commonly reported metrics include percentages of commands and barge-in, but the CSTR system did not support them.

(iii) *Task success*: This group of quantitative metrics includes task success and dialogue reward. Task success uses a binary approach, where each dialogue task is classified as successful if the user achieved the goal (e.g. booking a flight, hotel or car) as in Bohus and Rudnicky (2005). Dialogue reward combines task success and dialogue length in terms of system turns as in Lemon et al. (2006):

$$\text{DialogueReward} = \begin{cases} 100 - |\mathit{SystemTurns}|, & \text{for successful dialogue,} \\ 0 - |\mathit{SystemTurns}|, & \text{for failed dialogue.} \end{cases} \quad (11)$$

(iv) *User satisfaction*: These qualitative metrics include *easy to understand*, *system understood*, *task easy*, *interaction pace*, *what to say*, *system response*, *expected behaviour*, and *future use*. The sum of these metrics represents the overall user satisfaction score.

5.2. Experimental setup

Our experiments evaluated the three machine dialogue behaviours described in Sections 2 and 3—deterministic ('D'), fully-learnt ('F'), and semi-learnt ('S')—and were carried out with a user population of native speakers of English. Each user was presented with six dialogue tasks (travel bookings), with the system using each of the three behaviours twice, so that each user experienced all behaviours. The first three dialogues concerned single bookings and the last three dialogues concerned composite bookings. Table 5 shows examples of single and composite travel booking tasks. The six dialogues per user were collected using one of the following two sequences: DSFFSD and SDFFDS; i.e. half of the users interacted first with a deterministic behaviour, and the other half interacted first with a learnt behaviour. Whilst deterministic and semi-learnt behaviours started the dialogues interchangeably, fully-learnt behaviour always started the composite travel bookings. This sequence of dialogues was used because other alternative sequences such as {DSFFSD, DFSSFD, SDFFDS, SFDFDS, FSDDSF, FDSSDF} require larger data collections (the more data the more expensive and time-consuming). Each dialogue was logged using an extended version of the DATE dialogue annotation scheme (Walker and Passonneau, 2001). These log files were used to compute quantitative results. In addition, at the end of each dialogue, participants were asked to fill in a questionnaire (Table 6) in order to compute qualitative results, evaluated with a 5-point Likert scale, where 5 represents the highest score.

The set of 32 users voluntarily agreed to participate in the experimental evaluation. They had an average age of 36 with a gender distribution of 22 male (69%) vs. 10 female (31%). The participants' countries of origin were as follows: 17 from the UK (53%), 12 from USA (38%), and 3 from Canada (9%). From this user

Table 5
Sample travel booking tasks.

Booking	Task
Single	Try to book a *single* flight from *London* to *Paris* leaving on *December 6th* in the *afternoon*, and travelling with *any airline* What is the cost of the most expensive flight?
Composite	(a) Try to book a *return* flight from *Edinburgh* to *Amsterdam* leaving on *January 22nd* in the *morning*, and returning on the *1st of February* in the *evening*. What is the cost of the cheapest flight with *British Airways*? (b) Try to book a *cheap* hotel in *downtown* with *any hotel* brand . What is the cost of the cheapest hotel in downtown? (c) Try to rent a *compact car* near the *airport* for *three* *days* on *January 22nd* with pick-up time at *7PM* You don't have preference in rental company What is the rental cost of the most expensive car?

Table 6
Subjective measures for qualitative evaluation of human–machine task-oriented spoken dialogues, adapted from Walker et al. (2000).

Measure	Question
Easy to understand	Was the system easy to understand?
System understood	Did the system understand what you said?
Task easy	Was it easy to find the flight/hotel/car you wanted?
Interaction pace	Was the pace of interaction with the system appropriate?
What to say	Did you know what you could say at each point?
System response	Was the system fast and quick to reply to you?
Expected behaviour	Did the system work the way you expected it to?
Future use	Do you think you would use the system in the future?

population, 9 (28%) had no experience with spoken dialogue systems, 18 (56%) had some experience interacting with a spoken dialogue system at least once, and 5 (16%) were expert users. The latter were researchers in spoken dialogue processing (excluding the authors of this paper).

5.3. Experimental results

This subsection describes an analysis of results obtained from automatic and manual transcriptions at the syntactic and semantic level. Table 8 summarizes the results obtained using the three behaviours, using statistical significance tests to compare the semi-learnt behaviour against the deterministic and fully-learnt behaviours. For such a purpose data vectors (averaged per speaker) were verified through Lilliefors tests which indicated that they do not come from normal distributions. This suggests that non-parametric tests should be used. Thus, significance tests are reported with the Wilcoxon signed-rank test as suggested by Demsar (2006).

5.3.1. Dialogue efficiency

The fully-learnt behaviour seems to outperform significantly the other behaviours by obtaining fewer system turns, fewer user turns and less time. The result is something of an artifact, since the fully-learnt policy could induce infinite loops in some dialogue states. In this case the dialogues were manually stopped after three repetitive actions (considered as evidence of an infinite loop), leading to shorter dialogues but a lower task success. The purpose of testing this dialogue policy was threefold: (1) to evaluate how users perceive a dialogue policy with infinite loops; (2) to raise the issue of (in)coherent behaviour inferred by reinforcement learning agents, which has been ignored in previous related work; and (3) to compare its performance against a similar dialogue policy, but constrained with prior expert knowledge.

This phenomenon did not happen with deterministic or semi-learnt behaviours because their prior knowledge constrained more tightly the available actions per dialogue state. From these two dialogue strategies, it can be observed that semi-learnt behaviour outperformed deterministic, with significant differences in system and user turns. These results suggest that although learnt behaviours were optimized for dialogue efficiency, they cannot be evaluated in the same way. Therefore, a wider repertoire of evaluation metrics is preferable for a deeper analysis of dialogue behaviours.

5.3.2. Dialogue quality

Fully-learnt behaviour obtained the lowest word and keyword error rates. These results are not statistically significant, which suggests that behaviours were compared under similar recognition error rates. Deterministic and semi-learnt behaviours exhibited similar word and keyword error rates, but different event error rates. Typically, spoken dialogue strategies have to handle trade-offs between acceptance, confirmation and rejection of recognition hypotheses, which can be classified as correct or incorrect. Ideally, dialogue behaviours should choose actions maximizing correct acceptance/confirmation/rejection events, whilst minimizing the incorrect ones. From these results it can be inferred that semi-learnt behaviour handled such trade-offs better than deterministic behaviour by issuing more acceptances and fewer confirmations. This implies the use of more implicit

Table 7
Fragment of a real dialogue in the CSTR travel planning spoken dialogue system using fully-learnt action-selection following policies π_j^{*i}. The dialogue states are formed by the state variables shown in Table 4.

Agent	Policy	State	Action	Dialogue Act	Utterance
SYS	π_0^{*0}	0,0,0,0,0,0	gre	gre( )	Welcome to the CSTR travel planning system
		0,1,0,0,0,0	M_0^1		
	π_0^{*1}	1,0,0	M_0^2		
	π_0^{*2}	0,0,0,0,0	M_0^3		
	π_0^{*3}	0,0,0,0,0,0,0	req	req(DepCity)	Tell me your flight information
USR				pro(FlightType=single,DepCity=edinburgh, DesCity=paris)	*I would like a single flight from Edinburgh to Paris*
SYS	π_0^{*3}	0,2,2,0,0,2,2	acc		
	π_0^{*3}	2,2,2,0,0,2,2	mic	mic(FlightType=single,DepCity=edinburgh, DesCity=paris,Airline=bmi)+req(DepDate)	A single flight from Edinburgh to Paris travelling with BMI When do you want to travel?
USR				pro(Airline=air france)	*I would like to travel with Air France*
SYS	π_0^{*3}	2,4,4,0,0,2,4	sic	sic(Airline=air france)+req(DepDate)	A flight travelling with air france When do you want to travel?
USR				pro(DepDate=december sixth)	*December the sixth*
SYS	π_0^{*3}	2,4,4,2,0,4,4	acc		
SYS	π_0^{*3}	3,4,4,2,0,4,4	sic	sic(DepDate=may sixth)+req(DepTime)	A flight leaving on May sixth At what time do you prefer to travel?
USR				rep(DepDate=december sixth, DepTime=morning)	*Ehh I would like to travel on December the sixth in the morning*
.....					
SYS	π_0^{*3}	3,4,4,4,4,4,4			[terminal state]
	π_0^{*2}	0,0,2,2,0	dbq+sta	dbq(db)+sta(db)	Please wait while I retrieve information

confirmations and fewer explicit confirmations, which helps to explain why semi-learnt behaviour was more efficient than the deterministic one. Although dialogue policies were not optimized for 'event error rate' (see Eq. (12), p. 40), these results suggest that optimizing for dialogue efficiency produced an indirect optimization for such trade-offs. In addition, it can be observed that deterministic and semi-learnt behaviours are significantly different in all recognition events (correct/false acceptance/confirmation/rejection). In contrast, both learnt behaviours are significantly different in only half of the recognition events, suggesting that learnt behaviours act in a more similar way than deterministic behaviour.

5.3.3. Task success

Fully-learnt behaviour was significantly outperformed by the other behaviours that generated more successful conversations. This is where fully-learnt behaviour paid the price for generating some infinite dialogues that had to be artificially terminated before successful completion. In addition, whilst deterministic and semi-learnt behaviours were very similar in terms of task success, semi-learnt behaviour significantly outperformed its deterministic counterpart in terms of dialogue reward. This suggests that the dialogue reward metric is reflecting well the combined results from dialogue efficiency and dialogue accuracy.

5.3.4. User satisfaction

Users evaluated the semi-learnt behaviour as the best. Although, semi-learnt behaviour was significantly different to fully-learnt behaviour, it was not significantly different to its deterministic counterpart. A similar user satisfaction result was found by Singh et al. (2002) and Lemon et al. (2006). The performance of optimized confirmation strategies may be obscured by high recognition error rates. Future experiments could investigate optimized confirmation strategies under lower recognition error rates. In addition, the differences between learnt behaviours were statistically significant in the following qualitative metrics: *system understood*, *task easy*, *expected behaviour*, and *future use*. Similar differences were observed when comparing statistical sig-

Table 8
Results of the CSTR travel planning spoken dialogue system comparing three different dialogue behaviours, organized according to the following groups of metrics: dialogue efficiency, dialogue quality, task success and user satisfaction.

Measure	Behaviour			p-Values		
	Deterministic(1)	Fully-learnt(2)	Semi-learnt(3)	(1,2)	(1,3)	(2,3)
Avg. system turns	16.63	12.24	15.09	$\leqslant 0.05$	$\leqslant 0.05$	$\leqslant 0.05$
Avg. user turns	14.38	9.69	12.63	$\leqslant 0.05$	$\leqslant 0.05$	$\leqslant 0.05$
Avg. time (s)	177.23	139.59	165.11	$\leqslant 0.05$		
Word error rate	0.429	0.410	0.428			
Keyword error rate	0.300	0.278	0.301			
Event error rate	0.409	0.351	0.372			
Correct acceptance	5.51	26.34	20.95	$\leqslant 0.05$	$\leqslant 0.05$	
Correct confirmation	48.51	36.17	39.86	$\leqslant 0.05$	$\leqslant 0.05$	$\leqslant 0.05$
Correct rejection	5.18	2.37	1.92	$\leqslant 0.05$	$\leqslant 0.05$	
False acceptance	3.25	12.27	9.30	$\leqslant 0.05$	$\leqslant 0.05$	$\leqslant 0.05$
False confirmation	32.64	20.11	26.60	$\leqslant 0.05$	$\leqslant 0.05$	$\leqslant 0.1$
False rejection	4.91	2.55	1.36	$\leqslant 0.05$	$\leqslant 0.05$	
Avg. task success	0.94	0.62	0.95	$\leqslant 0.05$		$\leqslant 0.05$
Avg. dialogue reward	79.46	54.68	82.56	$\leqslant 0.05$	$\leqslant 0.05$	$\leqslant 0.05$
Easy to understand	4.34	4.31	4.44			
System understood	3.09	2.72	3.28	$\leqslant 0.05$		$\leqslant 0.05$
Task easy	3.50	3.00	3.45	$\leqslant 0.1$		$\leqslant 0.05$
Interaction pace	3.52	3.55	3.50			
What to say	3.45	3.47	3.58			
System response	3.67	3.64	3.63			
Expected behaviour	3.42	3.08	3.52	$\leqslant 0.05$		$\leqslant 0.05$
Future use	3.14	2.83	3.28	$\leqslant 0.05$		$\leqslant 0.05$
User satisfaction	28.14	26.59	28.67	$\leqslant 0.1$		$\leqslant 0.05$

(1) Note on statistical significance: typically, p-values $p \leqslant 0.05$ are considered to be statistically significant, and p-values $p \leqslant 0.1$ are indicative of a statistical trend.
(2) Note on task success: the drop of performance in fully-learnt behaviour was mainly caused by infinite loops, where the execution of action a in state s did not change the state $s' = s$.

nificance between deterministic and fully-learnt behaviour. These results suggest that those are the metrics with more impact on perceived system performance in the presence of unexpected dialogue behaviour such as infinite loops.

These results can be summarized as follows. First, dialogues by deterministic and semi-learnt behaviour were significantly more successful than dialogues by fully-learnt behaviour. These unsuccessful dialogues were reflected in the efficiency metrics, where fully-learnt behaviour falsely seems to be most efficient. Second, deterministic and semi-learnt behaviours are equally successful but the latter is more efficient (at $p \leqslant 0.05$ in system/user turns). Third, real users perceived fully-learnt behaviour as the worst (with statistical trend for deterministic vs. fully-learnt, and significant at $p \leqslant 0.05$ for fully-learnt vs. semi-learnt). Finally, the problem of infinite loops could have been avoided (e.g. by backing off from learnt behaviour to a deterministic one in dialogue states with potential infinite loops); however, if a dialogue policy uses fully-learnt behaviour without a good reward function or without constraints to generate dialogues that make sense to humans, then it may not learn *successful and coherent behaviours*. According to the quantitative and qualitative results above, it can be concluded that semi-learnt behaviour was better than the other behaviours.

5.4. Analysis of results based on users with only successful dialogues

A further (and possibly more fair) comparison of spoken dialogue behaviours was based on users with only successful dialogues (9 users out of 32, where each user did six dialogue tasks)—shown in Table 9. It shows a summary of results comparing deterministic, fully-learnt and semi-learnt behaviour; including statistical significance. Firstly, it can be observed that both learnt behaviours were more efficient (in system/user turns,

Table 9
Results of the CSTR travel planning spoken dialogue system using data from users—with only successful dialogues. They are organized in the following groups of metrics: dialogue efficiency, dialogue quality, task success and user satisfaction.

Measure	Behaviour			p-Values		
	Deterministic(1)	Fully-learnt(2)	Semi-learnt(3)	(1,2)	(1,3)	(2,3)
Avg. system turns	14.58	11.94	12.58	$\leqslant 0.05$	$\leqslant 0.05$	
Avg. user turns	12.50	9.75	10.23	$\leqslant 0.05$	$\leqslant 0.05$	
Avg. time (s)	159.74	142.69	132.48	$\leqslant 0.05$		
Word error rate	0.343	0.265	0.276			
Keyword error rate	0.209	0.137	0.167	$\leqslant 0.1$		
Event error rate	0.365	0.233	0.175		$\leqslant 0.1$	$\leqslant 0.1$
Avg. task success	1.00	1.00	1.00			
Avg. dialogue reward	85.42	88.06	87.42	$\leqslant 0.05$	$\leqslant 0.05$	
User satisfaction	31.28	31.78	32.39			

at $p \leq 0.05$) than their deterministic counterpart, and the differences between learnt behaviours were not statistically significant. Secondly, no significant differences were observed in dialogue quality. However, the statistical trend in event error rate suggests that the semi-learnt behaviour handled the trade-offs of acceptance/confirmation/rejection events more effectively. Thirdly, it can be noted that both learnt behaviours obtained more reward than their deterministic counterpart, and that therefore this metric is reflecting the significant differences observed from efficiency metrics. Last, similar to the results for all dialogues, the semi-learnt behaviour obtained the highest score in user satisfaction, but the differences were not statistically significant.

These results lead us to conclude that semi-learnt dialogue behaviour is a better alternative than deterministic, and indicate that its performance is comparable to that of fully-learnt behaviour when they are evaluated on only successful dialogues.

5.5. Do people want to talk to spoken dialogue systems?

At the end of each experimental session, participants were asked the following question: 'Would you use spoken dialogue systems for other tasks based on this experience?' Participants ranked their preference using a 5-point Likert scale, where the higher the score, the better the satisfaction. We observed that only 12% (4) percent of participants were pessimistic in their future use, 56% (18) of participants preferred to stay neutral, and 31% (10) were optimistic in its future use. The scores in preference of future use per user type were 3.0 for novice users, 3.28 for experienced users, and 3.2 for expert users (see p. 21 for proportions of user types). To further analyze this, consider splitting the group of participants: the first group with dialogue reward smaller than 80 and the rest in the second group. There was a 2.8 score in preference of future use for the first group of participants against a 3.7 score for the second group. Based on this result (significant at $p = 0.006$) it can be inferred that the higher the dialogue reward the higher the preference for future use of dialogue systems.

6. Evaluation of simulated behaviours

The evaluation of simulated behaviours described in this section have three purposes: (1) to investigate the differences between real and simulated speech recognition, (2) to investigate if simulated user behaviour generates user responses that resemble human responses, and (3) to investigate if the hand-crafted dialogue strategy is a reasonable baseline to compare against other competing dialogue strategies.

6.1. Real vs. simulated speech recognition

The real conversational environment used the ATK/HTK speech recognizer, and the simulated one used a simulated speech recognition error model (see Section 2.2). Recognition results in terms of Keyword Error

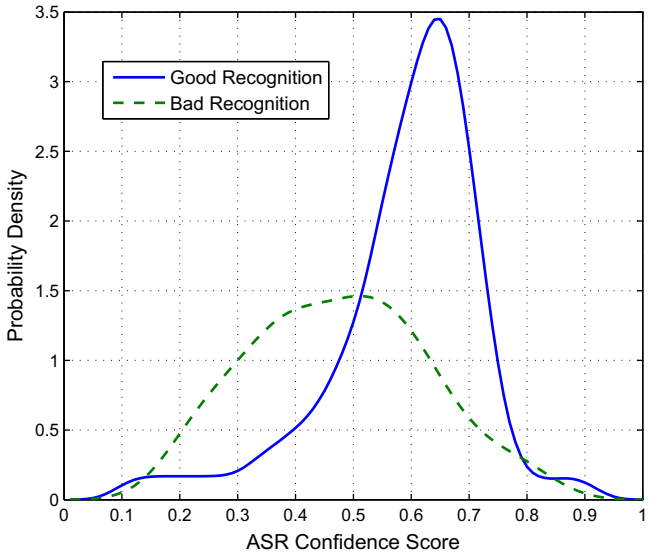

Fig. 12. Probability density functions estimated from observed speech recognition confidence scores of keywords in data collected by the CSTR travel planning system.

Rate (KER) for both environments were as follows: 20% in the simulated environment and 29% in the real one. For confidence scoring, the real environment showed confidence scores based on the probability density functions shown in Fig. 12 (estimated from real data based on a normal density function), and the simulated environment generated uniformly distributed random confidence scores resulting in equal numbers of confidence levels. It can be observed that simulation used a more conservative KER and different distributions of confidence levels. This is because no training data was assumed, where the realistic probability distributions for recognition errors and confidence scoring were unknown.

Previous work in Automatic Speech Recognition (ASR) simulation has assumed that exponential probability distributions can model the behaviour of ASR confidence scorers (Pietquin, 2004; Williams, 2006). This research found that this assumption does not hold for the ASR system used here. Instead, the *gamma probability distributions* are suggested to simulate ASR confidence scores, which are more flexible and include the exponential distribution. Thus, learnt dialogue policies in a second stage can be retrained with more realistic ASR behaviour in order to generate potentially even better policies. Nevertheless, it was found that even conservative ASR error modelling was sufficient to find better policies than deterministic behaviour.

6.2. Real vs. simulated user behaviour

Simulated user behaviour was compared against real user behaviour and against random user behaviour (see Section 2 for details on dialogue simulation). Because there is a variety of proposals on how to evaluate user simulations, we proposed two metrics to evaluate user behaviour based on dialogue similarity (using the Kulback–Leibler divergence) and dialogue coherence (using coherence error rate), and also validated their results with the more established Precision-Recall metric based on the *F*-measure score (Schatzmann et al., 2005). These metrics were applied as described in Appendix A. The objectives of this evaluation were: (a) to observe if the simulated user model used to learn the dialogue strategies was a reasonable thing to use and (b) to validate that dialogue realism could be distinguished by the proposed metrics (KL divergence and CER).

This evaluation used three sets of user responses: (1) real user responses were extracted from annotated data from the realistic environment, consisting in 192 dialogues including 4623 user utterances; (2) simulated coherent responses using Algorithm 1 described in Section 2; and (3) simulated random responses using the same algorithm, but with a random choice of user dialogue acts (at line 12) and with a random sequence of slots. All

Table 10
Evaluation of real and simulated user behaviour with Precision-Recall in terms of *F*-measure (the higher the better) and KL divergence (the lower the better).

Compared dialogues	*F*-measure		KL divergence
	Less strict	More strict	
Real1 vs. Real2	0.915	0.749	1.386
Real vs. simulated coherent	0.708	0.612	4.281
Real vs. simulated random	0.633	0.360	5.025
Simulated coherent vs. simulated random	0.417	0.247	6.532

Notes: (1) The less strict *F*-measure score considers a user response as a sequence of actions, and the more strict score considers a user response as a single action, (2) the real dialogues were divided into two subsets ('Real1' and 'Real2') to provide an upper-bound score, (3) KL divergence used Witten–Bell discounting to smooth the probability distributions.

user responses (real, simulated coherent or simulated random) were derived from machine dialogue acts in the real logged data, which enabled a fairer comparison. In addition, the user responses were not distorted because they were compared before speech recognition occurred.

Table 10 shows an evaluation of simulated user behaviour using Precision-Recall and KL divergence. It can be seen that both metrics agreed in the ranking of dialogue realism, including the proposed KL divergence metric. These results show that simulated coherent behaviour is more similar to real user behaviour than simulated random behaviour. It can be observed that the Precision-Recall of simulated coherent behaviour obtained higher scores than those reported before (Schatzmann et al., 2005; Georgila et al., 2006), approaching the upper-bound scores from real user behaviour.

In addition, the results in terms of Coherence Error Rate (CER) for real, simulated and random responses were 8.23%, 2.99%, 30.10%, respectively. It can be observed that simulated coherent behaviour behaved very optimistically, that is not very different from real user behaviour, and it is significantly different from the coherence of random behaviour. This metric is interesting because it evaluates a different perspective from the existing metrics, and so it may be used as a complementary evaluation.

6.3. Evaluating the baseline of machine dialogue behaviour

The use of speech recognition confidence scores forces spoken dialogue strategies to handle trade-offs among acceptance, confirmation and rejection events e_i, which can be classified as correct $E^c = \{ca, cc, cr\}$ or incorrect $E^f = \{fa, fc, fr\}$. Table 11 shows the categories of recognition events. A reasonable dialogue strategy would choose actions maximizing correct acceptance/confirmation/rejection events, whilst minimizing the incorrect ones. A simple metric to quantify these events is referred to as an *Event Error Rate (EvER)*:

$$EvER = \frac{count(e_i \in E^f)}{count(e_j \in \{E^c, E^f\})} \times 100. \quad (12)$$

For such a purpose, consider that speech recognition hypotheses fall within three equally distributed regions of confidence scores (assuming no training data): low confidence, medium confidence, and high confidence. In addition, consider the confirmation strategies Π of Table 12 for the three confidence regions. Which confirmation strategy is a better baseline of machine behaviour? For perfect speech recognizers it has to be 'Strategy1', because it leads to more efficient conversations in terms of number of system turns; but this is

Table 11
Speech recognition events in spoken dialogue systems.

Recognition event	Correct	False (or incorrect)
Acceptance	ca	fa
Confirmation	cc	fc
Rejection	cr	fr

Table 12
Confirmation strategies for different recognition confidence score regions. Notation: IC = implicit confirmations, EC = explicit confirmations, and AP = apologies.

Strategy	Low confidence	Medium confidence	High confidence
Strategy1	IC	IC	IC
Strategy2	EC	IC	IC
Strategy3	AP	EC	IC
Strategy4	AP	EC	EC
Strategy5	EC	EC	EC

Table 13
Event Error Rate (EvER) results of real dialogues for confirmation strategies of Table 12. Abbreviations: ca = correct acceptance, cc = correct confirmation, cr = correct rejection, fa = false acceptance, fc = false confirmation, and fr = false rejection.

Strategy	ca(%)	cc(%)	cr(%)	fa(%)	fc(%)	fr(%)	EvER(%)
Strategy1	73.6	0	0	26.3	0	0	26.3
Strategy2	71.9	2.2	0	17.0	9.3	0	26.3
Strategy3	26.7	44.6	9.3	2.5	14.4	2.2	19.2
Strategy4	0	71.4	9.3	0	17.0	2.2	19.2
Strategy5	0	73.6	0	0	26.3	0	26.3

unrealistic. Thus, a more reasonable choice of dialogue strategy is the one obtaining the lowest EvER score, and can be expressed as

$$\text{Baseline Strategy} = \arg\min_{\pi_i \in \Pi} EvER(\pi_i). \tag{13}$$

This metric was used to evaluate—with real data—the deterministic (hand-crafted) machine dialogue behaviour of the system, in order to find a reasonable baseline of machine dialogue behaviour. From the data collected by the system, we used all keywords (with automatic and manual transcriptions) including their corresponding speech recognition confidence scores, and computed the EvER for such confirmation strategies (Table 13). It can be seen that the deterministic behaviour of choice in this research (Strategy3) indeed obtained the lowest EvER, together with 'Strategy4'. Although they obtained the same result, the former is more attractive, due to its use of implicit confirmations because it leads towards more efficient conversations. Therefore, it can be concluded that the learnt dialogue strategies were compared against a reasonable baseline of deterministic machine dialogue behaviour.

7. Related work

7.1. Dialogue strategy learning

Walker (1993, 1996) developed the notion of a simulation environment to test dialogue strategies and Walker (1993) and Biermann and Long (1996) proposed the notion of automatically optimizing a dialogue strategy. Reinforcement learning approaches based on the Markov Decision Process (MDP) model were first applied to dialogue strategy learning by Levin and Pieraccini (1997) in a simulation-based approach and by Walker et al. (1998) in a corpus-based approach. Since then there have been several developments in the field (Levin et al., 2000; Walker, 2000; Young, 2000; Singh et al., 2002; Scheffler, 2002; Pietquin, 2004; Williams, 2006; Toney, 2007; Young et al., 2007; Frampton, 2008), largely adopting flat tabular reinforcement learning approaches. The scalability of such approaches is limited because of the curse of dimensionality: the exponential growth of the search space according to the number of state variables taken into account. Even a system with a simple state representation may have a large search space which can quickly become intractable. This problem has led to the use of function approximation (Denecke et al., 2004; Henderson et al., 2008) in order to find solutions on reduced state–action spaces. These investigations have been applied to small-scale dialogue systems aiming for a single global solution. However, less attention has been paid to finding solutions using a

divide-and-conquer approach: hierarchical POMDPs with a bottom-up approach have been applied to small state–action spaces (Pineau, 2004), and hierarchical reactive planning and learning has been used for dialogue systems with few slots of information (Lemon et al., 2006; Rieser and Lemon, 2008b). Our spoken dialogue system in the travel planning domain was implemented with five dialogue goals and 26 slots of information. This is the largest scale spoken dialogue system so far (in terms of dialogue goals and slots) tested using the reinforcement learning paradigm.

Our approach for incorporating prior expert knowledge into reinforcement learning agents is based on the Hierarchical Abstract Machines (HAMs) of Parr and Russell (1997). In this approach the system designer specifies a partial program (HAM) and leaves the unspecified part to the hierarchical reinforcement learning agent. This is an important extension to the fully-learnt approach because it constrains each hierarchical SMDP with some prior expert knowledge, in order to combine dialogue behaviour specified by human designers with behaviour automatically inferred by reinforcement learning agents.

Litman et al. (2000), Walker (2000), and Singh et al. (2002) incorporated prior knowledge into MDP-based spoken dialogue systems (NJFun, ELVIS) by means of hand-crafted rules used to compress the state–action space. This approach allowed them to perform very efficient dialogue strategy learning. However, NJFun and ELVIS do not provide a formal framework for incorporating prior knowledge and applying flat dialogue optimization. This is in contrast to our approach which is based on deterministic-stochastic finite state machines and adopts a hierarchical structure for optimization.

Heeman (2007) proposed combining the information-state update approach with reinforcement learning agents. In this approach the information-state (dialogue state) is hand-crafted by update rules based on preconditions and effects. A subset of preconditions that are easy to specify are hand-crafted, and those less easy to specify are left to the reinforcement learning agent. Again this uses flat reinforcement learning.

Williams (2008) proposed executing a POMDP and a hand-crafted dialogue controller in parallel. At each time step, the hand-crafted controller is in state s (e.g. semantic frame) and the POMDP is in belief state b (probability distribution over POMDP states), the hand-crafted controller nominates a subset of actions, and the POMDP updates a value function only for that particular subset of actions. Thus, a POMDP solution is found on a more compact space of policies. Our approach and Williams' approach share the idea of executing a partial program in parallel with an optimized decision-making model, but we use a decomposed MDP and optimize a hierarchy of partial programs, which is more scalable and suitable for reusability.

7.2. Evaluation of simulated user behaviour

Researchers in spoken dialogue tend to agree that realistic simulated user behaviour must exhibit 'human-like behaviour' (Georgila et al., 2006). Schatzmann et al. (2005) found that the quality of the learnt dialogue strategies is strongly dependent on the simulated user model, where good (realistic) user models help to find better policies than poor user models. Ai and Litman (2008) evaluated real and simulated dialogues using human judges and found that no strong agreement can be reached by humans on the quality of the dialogues, but humans consistently rank models of simulated user behaviour.

Previous work has proposed several evaluation metrics for assessing the realism of user simulations and can be grouped into two broad approaches: *dialogue similarity* and *system performance*. The former approach assumes that given a set of metrics, a set of simulated dialogues, and a set of real dialogues, the realism of simulated dialogues increases as their scores approach those obtained by real ones. Most previously proposed evaluation metrics fall within this approach (Eckert et al., 1997; Scheffler and Young, 2000; Scheffler and Young, 2002; Schatzmann et al., 2005; Filisko, 2006; Georgila et al., 2005; Cuayáhuitl et al., 2005; Rieser and Lemon, 2006; Ai and Litman, 2007). This approach is useful for giving a rough indication of the similarity between simulated and real dialogues, but it penalizes unseen behaviour (even when it may be realistic). The latter approach ranks simulated user models in terms of their prediction of the performance of a dialogue system. This is motivated by the fact that simulated user models should improve machine dialogue behaviours rather than generating human-like conversations (Williams, 2007). Both approaches are limited by the fact that they require real dialogue data, which may not exist at early stages of system development, and that they cannot distinguish if a given sequence of machine–user dialogue acts is realistic or not.

Our proposed KL divergence metric complements the previous dialogue similarity metrics by comparing probability distributions of user dialogue acts, and showed agreement with the Precision-Recall metric. In contrast with previously proposed metrics, our proposed coherence error rate metric can distinguish if a given sequence of machine–user dialogue acts is realistic or not. Because this latter metric uses hand-crafted coherence rules, a potential future work is to induce such rules automatically from data.

8. Conclusions and future work

In this paper we have developed a hierarchical reinforcement learning spoken dialogue system, based on an SMDP, and evaluated it with real users in a laboratory setting. Both fully-learnt and semi-learnt machine dialogue behaviours were used and compared with a baseline hand-crafted dialogue strategy. To the best of our knowledge, this is the first evaluation of SMDP-based reinforcement learning dialogue agents in a realistic environment.

Semi-learnt behaviour was quantitatively better than the other dialogue behaviours. It achieved similar task success to deterministic behaviour (95%) and more efficient conversations by using 9% fewer system turns, 12% fewer user turns, and 7% less time (at $p < 0.05$). It also outperformed fully-learnt behaviour by 35% in terms of higher task success (at $p < 0.05$); an evaluation based on users with only successful dialogues did not report significant differences. However, although fully-learnt behaviour resulted in inferior overall performance, it cannot be discarded as a better alternative to hand-crafted behaviour. But it is less flexible and less coherent than semi-learnt behaviour because it does not include a mechanism to guarantee coherent actions, which is essential for successful dialogues. On the other hand, while users did perceive significant qualitative differences between fully-learnt behaviour and the semi-learnt behaviour and statistical trend between fully-learnt and deterministic, users did not observe significant differences between deterministic and semi-learnt behaviours. Our key findings may be summarized as follows:

1. Hierarchical semi-learnt dialogue agents are a better alternative (with higher overall performance) than deterministic or fully-learnt behaviour.
2. The proposed simulated environment with coherent user behaviour, and distorted with conservative speech recognition error rates (keyword error rate of 20%) was sufficient for learning dialogue policies with superior performance to a reasonable hand-crafted behaviour.
3. The evaluation metrics Precision-Recall and KL divergence agreed in the ranking of dialogue realism.
4. Real users act with highly coherent behaviour at the dialogue act level (real users behaved coherently 92% of the time according to the metric 'coherence error rate').
5. Hierarchical reinforcement learning dialogue agents are feasible and promising for the (semi)automatic design of optimized behaviours in larger-scale spoken dialogue systems.

We suggest the following research avenues for endowing conversational agents with optimized, adaptive, robust, scalable and effective spoken dialogue behaviours.

First, one of the most important limitations of this work was the lack of a robust approach for updating slot values. Due to the fact that speech recognition hypotheses may include errors, it is not trivial to know when to update the recognized slot values and when to reject them. The effect of non-robust keyword updating is that the system eventually gives the impression of forgetting what has been said before. This highlights the importance of effective and efficient mechanisms for dialogue history tracking. Future research can incorporate beliefs into the knowledge rich-states of the proposed framework with ideas from approaches such as regression methods (Bohus and Rudnicky, 2005), POMDPs (Williams, 2006), or Bayesian updates (Thomson et al., 2008).

Second, we focused on optimizing confirmation strategies to keep their assessment simple rather than evaluating multiple dimensions. But there is a wide range of optimized behaviours that can be incorporated into this kind of system. For example: learning initiative strategies, learning to give help, learning to ground, learning to present information, learning to clarify, learning to negotiate, learning to recover from errors, learning multimodal strategies, and learning to collaborate. The thorough integration of all these behaviours into a single framework remains to be investigated.

Third, our optimization approaches include support for tabular hierarchical reinforcement learning. However, if a given subtask is intractable (i.e. the state–action space becomes too large and indecomposable) then alternative methods should be adopted to make such subtasks feasible. One of the most promising approaches reported in the literature of reinforcement learning is that of function approximation. The optimization approaches employed in this paper could be combined with function approximators such as neural networks or linear function approximation (Henderson et al., 2008).

Fourth, the current practice of reinforcement learning for spoken dialogue uses a single reward function. Although the proposed approaches in this paper allowed the use of a different reward function per subtask, the experimental setting used the same performance function across the entire hierarchy. Intuitively, hierarchical dialogue optimizations may require different types of reward function at different levels of granularity. Moreover, as the dialogue complexity increases, it becomes more difficult to specify such performance functions. It remains to be investigated how to specify or infer such hierarchical reward functions since the learnt behaviour strongly depends on the reward function (Walker, 1993, 2000; Rieser and Lemon, 2008a).

In the proposed approaches the system designer has manually to remove irrelevant state variables and actions for each subtask. Although this is useful because it allows the system designer to specify what to remove, it may become problematic if relevant information is removed, leading to unsafe state abstraction. Therefore, it would be useful to have a method for performing state abstraction of dialogue information in a safer way (Dietterich, 2000).

Finally, the simulated conversational environment that we used did not model errors as in a real environment, which was to be expected due to the lack of training data. Nonetheless, the experimental results provided evidence to conclude that this heuristic-based dialogue simulation approach was useful to learn dialogue strategies with superior performance compared with a reasonable baseline of deterministic behaviour. This result is relevant for spoken dialogue systems in new domains, where annotated dialogue data is not available. Our simulated environment could be enhanced with probability distributions estimated from real annotated data as in Schatzmann et al. (2007). However, due to the fact that collecting training data is costly and time-consuming, a potential future work is to investigate methods for generalizing simulated behaviours for dialogue systems across different domains.

Acknowledgements

Heriberto Cuayáhuitl was sponsored by PROMEP (http://promep.sep.gob.mx) and the Autonomous University of Tlaxcala (http://www.uatx.mx), and Oliver Lemon by EC FP6 "TALK". We thank Johanna Moore, Steve Young and the anonymous reviewers for their helpful comments on this work.

Appendix A. Evaluation metrics for user behaviour

The evaluation of real and simulated user behaviour was carried out with three evaluation metrics from two different perspectives. The first perspective evaluates 'dialogue similarity', where *Precision-Recall* quantifies the predicted user responses from a user simulation model (strongly penalizes unseen responses (Schatzmann et al., 2005)), and *probabilistic dialogue similarity* complements the previous metric by comparing probability distributions of user dialogue acts based on the Kulback–Leibler divergence (Cuayáhuitl et al., 2005). The second perspective evaluates 'dialogue coherence', where *coherence error rate* ignores the seen or unseen user responses; instead, it classifies them into coherent or incoherent responses.

To illustrate the evaluation metrics consider the following sub-dialogues with common system responses assumed from logged real data, but user responses may be real (as in Table 14) or simulated (as in Tables 15 and 16). The acronyms of dialogue act types are described in Table 1.

A.1. Precision-Recall

This measure is commonly used in the information retrieval field, and was suggested by Schatzmann et al. (2005) to evaluate how well a user simulation model can predict real user dialogue behaviour. *Precision* spec-

Table 14
Sample sub-dialogue with user responses assumed from logged real data.

Agent	Dialogue Act	Wording
SYS	gre(), req(dep_city)	Welcome to the travel planning system Tell me your flight information.
USR	pro(dep_city=amsterdam,flight_type=return)	*I would like a return flight leaving from Amsterdam*
SYS	sic(flight_type=return), req(dep_city)	A return flight, where are you leaving from?
USR	pro(dep_city=amsterdam)	*Amsterdam*

Table 15
Sample sub-dialogue with simulated coherent user responses.

Agent	Dialogue Act	Wording
SYS	gre(), req(dep_city)	Welcome to the travel planning system Tell me your flight information
USR	pro(dep_city=amsterdam,dep_time=morning, airline=air france, flight_type=return)	A return flight from Amsterdam travelling in the morning with Air France
SYS	sic(flight_type=return), req(dep_city)	A return flight, where are you leaving from?
USR	pro(dep_city=amsterdam)	*Amsterdam*

Table 16
Sample sub-dialogue with simulated random user responses.

Agent	Dialogue Act	Wording
SYS	gre(), req(dep_city)	Welcome to the travel planning system Tell me your flight information
USR	con(dest_city=yes)	*Yes*
SYS	sic(flight_type=return), req(dep_city)	A return flight, where are you leaving from?
USR	pro(des_city=amsterdam)	*To Amsterdam*

ifies the fraction of correctly predicted real user responses from all simulated responses. *Recall* specifies the fraction of correctly predicted real user responses from all real responses. They are expressed as

$$\text{Precision} = \frac{\text{Number of correctly predicted user responses}}{\text{Total number of simulated user responses}} \tag{14}$$

and

$$\text{Recall} = \frac{\text{Number of correctly predicted user responses}}{\text{Total number of real user responses}}. \tag{15}$$

These scores are interpreted as the higher the more realistic the user responses. An average score of recall (R) and precision (P) called F-measure is defined by

$$F = \frac{2PR}{(P+R)}. \tag{16}$$

If we want to compute the F-measure score in dialogue data, the slot values can be ignored to reduce data sparsity while preserving the conveyed information. Schatzmann et al. (2005) suggested to compute Precision-Recall by considering a user dialogue act as a sequence of actions, e.g. the dialogue act 'pro(dep_city, flight_type)' is equivalent to {pro(dep_city), pro(flight_type)}. Considering the given sample sub-dialogues, the F-measure score for real vs. simulated coherent responses is $F = 0.75$, and the score for real vs. simulated random responses is $F = 0$. Alternatively, the scores can be computed in a more strict way by considering each

user response as a single user action instead of multiple ones. Precision-Recall can be recomputed as follows: the scores for real vs. simulated coherent responses are $F = 0.5$; and the score for real vs simulated random responses is $F = 0$.

A.2. Probabilistic dialogue similarity

The purpose of this measure is to evaluate the probabilistic similarity between two sets of dialogues. The similarity between real and simulated dialogues has been analyzed using the Kulback–Leibler divergence (Cuayáhuitl et al., 2005), and here we propose to apply it in a simpler way. First, compute two smoothed probability distributions of machine–user dialogue acts, without slot values for reduced combinations: P for one data set and Q for the other. For example: P represents a distribution of the set of real dialogues and Q a distribution of the set of simulated ones. Then compute the symmetric distance according to

$$D(P,Q) = \frac{D_{KL}(P\|Q) + D_{KL}(Q\|P)}{2}, \tag{17}$$

where D_{KL} is the Kulback–Leibler divergence (distance) between P and Q:

$$D_{KL}(P\|Q) = \sum_i p_i \log_2 \left(\frac{p_i}{q_i}\right). \tag{18}$$

Tables 17 and 18 use the sample sub-dialogues of this subsection in order to show the divergence between real and simulated coherent user responses, and between real and simulated random user responses. The probability distributions of occurrence P and Q were smoothed by assigning a probability mass of 0.1 to unseen events, and the method of preference can be used to address the issue of data sparsity. It can be observed that the symmetric divergence between real and simulated random user responses (2.536) is greater than between real and simulated coherent ones (0.759). This reflects the intuitive perception that the more realistic the user responses, the shorter the divergence.

It can be observed that this metric gives the same ordering on user simulations than the Precision-Recall metric. A validation of this ordering based on a corpus of real human–machine dialogues is reported in Section 6.

Table 17
Dialogue similarity results for real vs. simulated coherent sub-dialogues.

Dialogue Act Pairs (SYS:USR)	P	Q	$D_{KL}(P\|Q)$	$D_{KL}(Q\|P)$
gre(),req(dep_city):pro(dep_city,flight_type)	0.45	0.45	0.000	0.000
sic(flight_type)+req(dep_city):pro(dep_city)	0.45	0.10	0.976	−0.217
sic(flight_type)+req(dep_city):pro(dep_city,des_city,dep_time,airline)	0.10	0.45	−0.217	0.976
Divergence			0.759	0.759

Table 18
Dialogue similarity results for real vs. simulated random sub-dialogues.

Dialogue Act Pairs (SYS:USR)	P	Q	$D_{KL}(P\|Q)$	$D_{KL}(Q\|P)$
gre(),req(dep_city):pro(dep_city,flight_type)	0.45	0.05	1.426	−0.158
sic(flight_type)+req(dep_city):pro(dep_city)	0.45	0.05	1.426	−0.158
gre(),req(dep_city):con(des_city)	0.05	0.45	−0.158	1.426
sic(flight_type)+req(dep_city):pro(des_city)	0.05	0.45	−0.158	1.426
Divergence			2.536	2.536

Table 19
Results of coherence for real and simulated user responses.

Data set	Dialogue Act Pairs (SYS:USR)	$incoherent(a_t^u, k_t^u)$
Real	gre(),req(dep_city):pro(dep_city)	0
	gre(),req(dep_city):pro(flight_type)	0
	sic(flight_type),req(dep_city):pro(dep_city)	0
Simulated coherent	gre(),req(dep_city):pro(dep_city)	0
	gre(),req(dep_city):pro(dep_time)	0
	gre(),req(dep_city):pro(airline)	0
	gre(),req(dep_city):pro(flight_type)	0
	sic(flight_type),req(dep_city):pro(dep_city)	0
Simulated random	gre(),req(dep_city):con(dest_city)	1
	sic(flight_type),req(dep_city):pro(des_city)	0

A.3. Coherence Error Rate

An evaluation metric called *Coherence Error Rate (CER)* is proposed due to the fact that most previously used metrics penalize unseen user responses even when they may be realistic. The key assumption in this metric is that given a user knowledge-base k_t^u and a set of dialogue coherence rules encoded into a function, we can evaluate—in an approximated form—whether a user action a_t^u is coherent or not. This metric rates errors (in this context, incoherent dialogue acts) from a set of observed events (user dialogue acts in the data), in terms of dialogue act types (see Table 1):

$$CER = \frac{\sum incoherent(a_t^u, k_t^u)}{count(a_t^u)} \times 100, \tag{19}$$

where the coherence of user dialogue acts is evaluated according to

$$incoherent(a_t^u, k_t^u) = \begin{cases} 0, & \text{if } a_t^u \in \{pro, rep\} \text{ and unconfirmed slot in focus in } k_t^u, \\ 0, & \text{if } a_t^u \in \{con\} \text{ and } a_t^m \in \{sec, mec\}, \\ 0, & \text{if } a_t^u \in \{pro, rep\} \text{ and } a_t^m \in \{rel\}, \\ 1, & \text{otherwise.} \end{cases} \tag{20}$$

Eq. (20) is suited for simple slot-filling applications, but for more complex dialogues more rules have to be added. This metric takes into account user dialogue acts and decomposes them into dialogue acts with a single slot and without slot value, e.g. $pro(des\_city)$. This procedure incorporates the conveyed information, and assumes that the slot values are always consistent given a user goal at the beginning of the conversation. In addition, this evaluation metric considers user responses with silences or incomplete dialogue acts as incoherent, the explanation for this consideration is because whatever the user said (e.g. mumbles or out-of-vocabulary words), it was not possible to extract a user dialogue act contributing to the conversation (Table 19).

Given the sample sub-dialogues of this subsection, Table 19 shows the results of coherence for real, simulated coherent and simulated random user responses: 0%, 0%, 50%, respectively. Note that although simulated coherent user responses do not match the real ones, they are not being penalized because they are responses that make sense according to the dialogue history.

References

Ai, H., Litman, D., 2007. Knowledge consistent user simulations for dialogue systems. In: INTERSPEECH, pp. 2697–2700.

Ai, H., Litman, D., 2008. Assessing dialog system user simulation evaluation measures using human judges. In: Association for Computational Linguistics (ACL), pp. 622–629.

Barto, A., Mahadevan, S., 2003. Recent advances in hierarchical reinforcement learning. Discrete Event Dynamic Systems: Theory and Applications 13, 41–77.

Biermann, A., Long, P., 1996. The composition of messages in speech-graphics interactive systems. In: International Symposium on Spoken Dialogue, pp. 97–100.

Bohus, D., Rudnicky, A., 2005. Sorry, I didn't catch that! – an investigation of non-understanding errors and recovery strategies. In: Workshop on Discourse and Dialogue (SIGDIAL), Lisbon, Portugal, pp. 128–143.

Bohus, D., Rudnicky, A., 2005. Constructing accurate beliefs in spoken dialogue systems. In: IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU), San Juan, Puerto Rico, pp. 272–277.

Cuayáhuitl, H., Renals, S., Lemon, L., Shimodaira, H., 2005. Human–computer dialogue simulation using hidden Markov models. In: IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU), San Juan, Puerto Rico, pp. 290–295.

Cuayáhuitl, H., Renals, S., Lemon, L., Shimodaira, H., 2006. Reinforcement learning of dialogue strategies using hierarchical abstract machines. In: IEEE Workshop on Spoken Language Technology (SLT), Palm Beach, Aruba, pp. 182–185.

Cuayáhuitl, H., Renals, S., Lemon, L., Shimodaira, H., 2007. Hierarchical dialogue optimization using semi-Markov decision processes. In: INTERSPEECH, Antwerp, Belgium, pp. 2693–2696.

Cuayáhuitl, H., 2009. Hierarchical Reinforcement Learning for Spoken Dialogue Systems. Ph.D. Thesis, University of Edinburgh.

Cheyer, A., Martin, D., 2001. The open agent architecture. Journal of Autonomous Agents and Multi-Agent Systems 4, 143–148.

Denecke, M., Dohsaka, K., Nakano, M., 2004. Fast reinforcement learning of dialogue policies using stable function approximation. In: International Joint Conference on Natural Language Processing (IJCNLP), Jeju, Korea, pp. 1–11.

Demsar, J., 2006. Statistical comparisons of classifiers over multiple data sets. Journal of Machine Learning Research 7, 1–30.

Dietterich, T., 2000. Hierarchical reinforcement learning with the MAXQ value function decomposition. Journal of Artificial Intelligence Research 13 (1), 227–303.

Dietterich, T., 2000. An overview of MAXQ hierarchical reinforcement learning. In: Symposium on Abstraction, Reformulation, and Approximation (SARA), HorseshoeBay, TX, USA, pp. 26–44.

Eckert, W., Levin, E., Pieraccini, R., 1997. User modeling for spoken dialogue system evaluation. In: IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU), Santa Barbara, CA, pp. 80–87.

Filisko, E., 2006. Developing Attribute Acquisition Strategies in Spoken Dialogue Systems Via User Simulation. Ph.D. Thesis, Massachusetts Institute of Technology.

Frampton, M., 2008. Using Dialogue Acts in Dialogue Strategy Learning: Optimizing Repair Strategies. Ph.D. Thesis, University of Edinburgh.

Georgila, K., Henderson, J., Lemon, O., 2005. Learning user simulations for information state update dialogue systems. In: INTERSPEECH, Lisbon, Portugal, pp. 893–896.

Georgila, K., Lemon, O., Henderson, J., 2005. Automatic annotation of communicator dialogue data for learning dialogue strategies and user simulations. In: Workshop on the Semantics and Pragmatics of Dialogue (Brandial), DIALOR, Nancy, France.

Georgila, K., Henderson, J., Lemon, O., 2006. User simulation for spoken dialogue systems: learning and evaluation. In: INTERSPEECH, Pittsburgh, USA, pp. 267–659.

Grosz, B., Sidner, C., 1986. Attention, intentions and the structure of discourse. Computational Linguistics 12 (3), 175–204.

Heeman, P., 2007. Combining reinforcement learning with information-state update rules. In: Human Language Technology Conference (HLT), Rochester, NY, USA, pp. 268–275.

Henderson, J., Lemon, O., Georgila, K., 2008. Hybrid reinforcement/supervised learning of dialogue policies from from fixed data sets. Computational Linguistics 34, 487–511.

Kaelbling, L., Littman, M., Moore, A., 1996. Reinforcement learning: a survey. Journal of Artificial Intelligence Research 4, 237–285.

Lemon, O., Georgila, K., Henderson, K., 2006. Evaluating effectiveness and portability of reinforcement learned dialogue strategies with real users: the TALK TownInfo evaluation. In: IEEE Workshop on Spoken Language Technology (SLT), Palm Beach, Aruba, pp. 178–181.

Lemon, O., Liu, X., Shapiro, D., Tollander, C., 2006. Hierarchical reinforcement learning of dialogue policies in a development environment for dialogue systems: REALL-DUDE. In: Workshop on the Semantics and Pragmatics of Dialogue (BRANDIAL), Postdam, Germany, pp. 185–186.

Levin, E., Pieraccini, R., 1997. A stochastic model of computer–human interaction for learning dialog strategies. In: Europeach Conference on Speech Communication and Technology (Eurospeech), Rhodes, Greece, pp. 1883–1886.

Levin, E., Pieraccini, R., Eckert, W., 2000. A stochastic model of human machine interaction for learning dialog strategies. IEEE Transactions on Speech and Audio Processing 8 (1), 11–23.

Litman, D., Allen, J., 1987. A plan recognition model for subdialogues in conversations. Cognitive Science 11, 163–200.

Litman, D., Kearns, M., Singh, S., Walker, M., 2000. Automatic optimization of dialogue management. In: International Conference on Computational Linguistics (COLING), Saarbrucken, Germany, pp. 501–508.

Marthi, B., Russell, S., Andre, D., 2006. A compact, hierarchical Q-function decomposition. In: Conference on Uncertainty in Artificial Intelligence (UAI), Cambridge, MA, USA.

Paek, T., Pieraccini, R., 2008. Automating spoken dialogue management design using machine learning: an industry perspective. Speech Communication 50, 716–729.

Parr, R., Russell, S., 1997. Reinforcement learning with hierarchies of machines. In: Neural Information Processing Systems Conference (NIPS), Denver, CO, USA, pp. 1043–1049.

Pietquin, O., 2004. A Framework for Unsupervised Learning of Dialogue Strategies. Ph.D. Thesis, Faculté Polytechnique de Mons.

Pineau, J., 2004. Tractable Planning Under Uncertainty: Exploiting Structure. Ph.D. Thesis, Carnegie Mellon University.

Rieser, V., Lemon, O., 2006. Cluster-based user simulations for learning dialogue strategies. In: INTERSPEECH, Pittsburgh, USA, pp. 1766–1769.

Rieser, V., Lemon, O., 2008a. Automatic learning and evaluation of user-centered objective functions for dialogue system optimization. In: International Language Resources and Evaluation (LREC), Marrakech, Morocco, pp. 2536–2361.

Rieser, V., Lemon, O., 2008b. Learning effective multimodal dialogue strategies from Wizard-Of-Oz data: bootstrapping and evaluation. In: Association for Computational Linguistics (ACL), Columbus, OH, USA, pp. 638–646.

Rieser, V., 2008. Bootstrapping Reinforcement Learning-based Dialogue Strategies from Wizard-Of-Oz Data. Ph.D. Thesis, Saarland University.
Roy, N., Pineau, J., Thrun, S., 2000. Spoken dialogue management using probabilistic reasoning. In: Association for Computational Linguistics (ACL), Hong Kong, pp. 93–100.
Russell, S., Norvig, P., 2003. Artificial Intelligence: A Modern Approach. Pearson Education.
Singh, S., Litman, D., Kearns, M., Walker, M., 2002. Optimizing dialogue management with reinforcement learning: experiments with the NJFun system. Journal of AI Research 16, 105–133.
Schatzmann, J., Georgila, D., Young, S., 2005. Quantitative evaluation of user simulation techniques for spoken dialogue systems. In: Workshop on Discourse and Dialogue (SIGDIAL), Lisbon, Portugal, pp. 45–54.
Schatzmann, J., Stuttle, M.N., Weilhammer, K., Young, S., 2005. Effects of the user model on simulation-based learning of dialogue strategies. In: IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU), San Juan, Puerto Rico, pp. 220–225.
Schatzmann, J., Thomson, B., Young, S., 2007. Error simulation for training statistical dialogue systems. In: IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU), Kyoto, Japan, pp. 526–531.
Scheffler, K., Young, S., 2000. Probabilistic simulation of human–machine dialogues. In: International Conference on Acoustics, Speech, and Signal Processing (ICASSP), Istanbul, Turkey, pp. 1217–1220.
Scheffler, K., Young, S., 2002. Automatic learning of dialogue strategy using dialogue simulation and reinforcement learning. In: Human Language Technology Conference (HLT), San Diego, CA, USA, pp. 12–19.
Scheffler, K., 2002. Automatic Design of Spoken Dialogue Systems. Ph.D. Thesis, Cambridge University.
Seneff, S., Hurley, E., Lau, R., Pao, C., Schmid, P., Zue, V., 1998. Galaxy-II: a reference architecture for conversational system development. In: ICSLP, Sydney, Australia, pp. 931–934.
Sutton, R., Barto, A., 1998. Reinforcement Learning: An Introduction. MIT Press.
Toney, D., 2007. Evolutionary Reinforcement Learning of Spoken Dialogue Strategies. Ph.D. Thesis, University of Edinburgh.
Thomson, B., Schatzmann, J., Young, S., 2008. Bayesian update of dialogue state for robust dialogue systems. In: International Conference on Acoustics, Speech, and Signal Processing (ICASSP), Las Vegas, USA.
Walker, M., 1993. Informational Redundancy and Resource Bounds in Dialogue. Ph.D. Thesis, University of Pennsylvania.
Walker, M., 1996. The effect of resource limits and task complexity on collaborative planning in dialogue. Artificial Intelligence 85, 181–243.
Walker, M., Fromer, J., Narayanan, S., 1998. Learning optimal dialogue strategies: a case study of a spoken dialogue agent for email. In: Association for Computational Linguistics (ACL), Montreal, Canada, pp. 1345–1351.
Walker, M., 2000. An application of reinforcement learning to dialogue strategy selection in a spoken dialogue system for email. Journal of Artificial Intelligence Research 12, 387–416.
Walker, M., Kamm, C., Litman, D., 2000. Towards developing general models of usability with PARADISE. Natural Language Engineering 6, 363–377.
Walker, M., Passonneau, R., 2001. DATE: a dialogue act tagging scheme for evaluation of spoken dialogue systems. In: Human Language Technology Conference (HLT), San Diego, CA, USA, pp. 1–8.
Ward, W., 1994. Extracting information from spontaneous speech. In: International Conference on Speech and Language Processing (ICSLP), Yokohama, Japan, pp. 18–22.
Williams, J., 2006. Partially Observable Markov Decision Processes for Spoken Dialogue Management. Ph.D. Thesis, Cambridge University.
Williams, J., Young, S., 2007. Partially observable Markov decision processes for spoken dialog systems. Computer Speech and Language 21 (2), 393–422.
Williams, J., 2007. A method for evaluating and comparing user simulations: the Cramer–Von Misses divergence. In: IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU), Kyoto, Japan.
Williams, J., 2008. The best of both worlds: unifying conventional dialog systems and POMDPs. In: INTERSPEECH, Brisbane, Australia.
Yamagishi, J., Zen, H., Toda, T., Tokuda, K., 2007. Speaker-independent HMM-based speech synthesis system – HTS-2007 system for the Blizzard challenge 2007. In: The Blizzard Challenge, Bonn, Germany.
Young, S., 2000. Probabilistic methods in spoken dialogue systems. Philosophical Transactions of the Royal Society (Series A) 358, 1389–1402.
Young, S., 2006. The HTK Book. Cambridge University Engineering Department.
Young, S., 2007. ATK: An Application Toolkit for HTK. Cambridge University Engineering Department.
Young, S., Schatzmann, J., Weilhammer, K., Ye, H., 2007. The hidden information state approach to dialogue management. In: International Conference on Acoustics, Speech, and Signal Processing (ICASSP), Honolulu, Hawaii, pp. 149–152.