# Adaptive Natural Language Generation in Dialogue using Reinforcement Learning

**Oliver Lemon**
School of Informatics
Edinburgh University
olemon@inf.ed.ac.uk
http://homepages.inf.ed.ac.uk/olemon

## Abstract

This paper presents a new model for adaptive Natural Language Generation (NLG) in dialogue, showing how NLG problems can be approached as statistical planning problems using Reinforcement Learning. This approach brings a number of theoretical and practical benefits such as fine-grained adaptation, generalization, and automatic (global) optimization. We present the model and related work in statistical/trainable NLG, discuss its applications, and provide a demonstration of the approach, showing policy learning for adaptive information presentation decisions (Contrast, Cluster, or List items). An adaptive NLG policy learned in our framework shows a statistically significant 27% relative increase in reward over an "RL-majority" baseline policy for the same task. We thereby also show that that such NLG problems should be approached in combination with dialogue management decisions, and we show how to jointly optimize NLG and dialogue management plans.

## 1 Introduction

Natural Language Generation (NLG) in dialogue is often characterised as choosing "how" to say something once "what to say" has been determined. In principle, NLG in dialogue thus comprises a wide variety of decisions, ranging over content structuring, choice of referring expressions, use of ellipsis, aggregation, and choice of syntactic structure, to choice of intonation markers. In computational dialogue systems, "what to say" is usually determined by a dialogue manager (DM) component, via planning, hand-coded rules, finite state machines, or learned policies, and "how to say it" is then very often defined by simple templates or hand-coded rules which define appropriate word strings to be sent to a speech synthesizer or screen.

Previous statistical approaches to NLG are reviewed in section 2, but none of them have explored NLG as statistical *planning*. Some aspects of NLG have been treated as planning (Koller and Stone, 2007; Stone et al., 2003), but not statistically. Prior dialogue-related work in statistical planning (e.g. Reinforcement Learning) has dealt only with policies for planning dialogue acts in an information gathering phase, and has not been applied to NLG decisions themselves (though see Rieser and Lemon (2008) for work in multimodal generation).

Learning approaches have several key potential advantages over template-based and rule-based approaches to NLG (we discuss "trainable" NLG in section 2.1) in dialogue systems:

- ability to adapt to fine-grained changes in dialogue context
- data-driven development cycle, with reduced development costs for industry.
- provably optimal action policies with a precise mathematical model for action selection
- ability to generalize to unseen dialogue states

We aim to illustrate these advantages in the demonstration system described in this paper.



*Proceedings of the 12th Workshop on the Semantics and Pragmatics of Dialogue, June 2–4, 2008, London, U.K.*

## 2 Prior work

There are 3 main approaches to generating system utterances in dialogue systems: template-based NLG, conventional NLG as developed in the text generation literature (Reiter and Dale, 2000), and more recently, trainable generation (Barzilay and Lapata, 2005; Duboue and McKeown, 2003; Stent et al., 2004; Walker et al., 2001; Walker et al., 2007).

*Template-based* generation is typically used in industrial dialogue systems, and even in most state-of-the-art research systems. However, this approach requires that new templates be created by hand for each application, and severely limits that system's ability to adapt to dialogue context or user preferences, due to the practical constraints of having to write different templates for each possible combination of feature values.

*Conventional NLG* typically follows a pipeline architecture consisting of three main modules (Reiter and Dale, 2000):

- (i) a text planner which performs content selection and discourse structuring,
- (ii) a sentence planner which selects attributes for referring expressions, aggregates content into sentence-size units, and selects lexical items, and
- (iii) a surface realizer which converts sentence plans into natural language.

This approach has been successfully applied in systems that tailor their presentations to the user's preferences (Carenini and Moore, 2006; Demberg and Moore, 2006; Moore et al., 2004; Walker et al., 2004) and to the dialogue context (Isard et al., 2003), but these systems generally use rules that are specifically hand-crafted for a particular domain. A major problem with these standard NLG approaches is that hand-coded rules, manually-set thresholds, and templates all severely limit the adaptivity that can be achieved in NLG, both in the amount of adaptivity possible and the ability to adapt to fine-grained changes in the dialogue context, user behaviour, or environment (e.g. noise levels). The standard approaches are limited by the expertise of the system designer, and the adaptivity that they can encode in their rules or templates. Statistical approaches in general promise a more practical, effective, and theoretically well-founded approach to adaptivity in NLG, because they are not limited by human design capacities, and can be trained from data. Conventional NLG approaches can also be too slow for real-time dialogue applications (Stent et al., 2004). There has therefore been recent interest in statistical methods in the area of "trainable" NLG.

### 2.1 Trainable NLG

*Trainable NLG* is a more recent approach, where automatic techniques are used to train NLG modules, or to adapt them to specific domains and/or types of user. However, this work has focussed on local optimization through supervised learning, and has not explored global decision-theoretic planning approaches such as Reinforcement Learning.

Early work here focused on supervised learning of how to produce surface forms from sentence plans, using overgeneration and ranking, using either bigram language models (Oh and Rudnicky, 2002), or ranking rules learned from a corpus of manually ranked training examples (Walker et al., 2001). More recent work has extended this approach to sentence-planning (Stent et al., 2004).

In this work, given a content plan (the propositions to express and discourse relations among them), a generator first produces a set of text-plan trees, consisting of speech acts to be communicated, and the rhetorical relations between them. For each of these, a set of candidate sentence plans are generated by a heuristically ordered set of clause-combining operations. The sentence plan ranker is then trained by using the RankBoost algorithm to learn a set of ranking rules from a manually labelled set of examples.

For content selection, recent research has shown that given a corpus of texts and the database of facts or events it describes, content selection rules can be learned (Barzilay and Lapata, 2005; Duboue and McKeown, 2003). In this work, content selection has been treated as a binary classification task. Here, semantic units in the database are first aligned with sentences in the corpus, and then classification is used to learn whether or not a semantic unit should be included in the text.

However, as explained above, these types of supervised learning used for NLG do not model the

required optimization and planning of sequences of actions-in-context which we propose to capture with RL techniques. An interesting issue for future work is how these types of classifier-based learning can be integrated with the MDP approach propsosed here.

## 3 The model: NLG as statistical planning

This paper treats NLG as a statistical planning and optimization problem using decision theory in the framework of Markov Decision Processes (MDPs), similar to (Rieser and Lemon, 2008). The main advance here is to treat aspects of NLG within the same MDP-based planning and learning frameworks as have been successfully applied in speech recognition and dialogue management, for example (Levin and Pieraccini, 1997; Walker et al., 1998; Singh et al., 2002; Young, 2000).

We now propose to model the NLG problem as a Markov Decision Process (MDP). Here a stochastic system interacting with its environment (in our case, the user of the dialogue system) through its actions is described by a number of states $\{s_i\}$ in which a given number of actions $\{a_j\}$ can be performed. In a dialogue system, the states represent the possible dialogue contexts (e.g. how much information we have so far obtained from the user) and the actions are now system dialogue and NLG actions.

Each state-action pair is associated with a transition probability $\mathcal{T}_{ss'}$: the probability of moving from state $s$ at time $t$ to state $s'$ at time $t+1$ after having performed action $a$ when in state $s$. This transition is also associated with a reinforcement signal (or reward) $r_{t+1}$ describing how good the result of action $a$ was when performed in state $s$. In dialogue these reward signals are most often associated with task completion and dialogue length, but we will also associate them with NLG decisions.

To control a system described in this way, one then needs a strategy or policy $\pi$ mapping all states to actions: $\pi(s) = P(a|s)$ In this framework, a Reinforcement Learning agent is a system aiming at optimally mapping states to actions, i.e. finding the best strategy so as to maximize an overall reward $R$ which is a function (most often a weighted sum) of all the immediate rewards. In dialogue (and many other problems) the reward for an action is often not immediate, but is *delayed* until successful completion of a task. In the most challenging cases, actions may affect not only immediate reward, but also the next situation and, via that, all subsequent rewards.

In general, then, we are trying to find an action policy $\pi$ which maximises the value $Q^\pi(s,a)$ of choosing action $a$ in state $s$, which is given by the Bellman equation:

$$Q^\pi(s,a) = \sum_{s'} \mathcal{T}_{ss'}[\mathcal{R}_{ss'} + \gamma V^\pi(s')] \qquad (1)$$

(Here we denote the expected immediate reward by $\mathcal{R}_{ss'}$, $\gamma$ is a discount factor between 0 and 1, $V^\pi(s)$ is the value of state $s'$ according to $\pi$, see (Sutton and Barto, 1998)).

If the transition probabilities are known, an analytic solution can be computed by dynamic programming. Otherwise the system has to learn the optimal strategy by a trial-and-error process, for example using Reinforcement Learning methods (Sutton and Barto, 1998) as we do in this paper. Trial-and-error search and delayed rewards are the two main features of Reinforcement Learning.

In prior work on dialogue strategy learning, only dialogue acts (e.g. greet, ask slot, explicit confirm) chosen by the system have been optimized. Here we go beneath the level of dialogue acts to plan NLG actions such as content structuring. Several key questions thus arise – how to represent different NLG actions for planning, what context features and state representations are important in the MDP, and what reward signals can be used to optimize NLG?

We now present a fully worked example to show the model in use.

## 4 Learning for Adaptive Information Presentation

One of the classic problems in NLG is how to present one or more items to a user, for example by simply listing them, contrasting them in respect of some attributes, or clustering similar items together (e.g. (Moore et al., 2004; Walker et al., 2007)). For example the system may present items like so:

- LIST: "There are four hotels meeting your criteria. The first is the Royal, the second is ..."

- CONTRAST: "The Oak is an expensive central hotel. The Royal is cheap but is not central. . . ."
- CLUSTER: "There are 7 expensive hotels and 11 cheap ones, . . .".

We will model these decisions in an MDP, and solve it using trial-and-error exploration, using Reinforcement Learning methods. First, we model the states of the system.

### 4.1 State space

In this example we will have 3 search constraint slots that the user can fill (for instance `food type, location, price range` for a restaurant search application, or `artist, album, genre` for music browsing). These slots can be either filled or confirmed. Confirmed slots have 100% chance of being correct, and filled slots only 80% chance, thereby modeling noise in the speech recognition environment (see also Lemon and Liu (2007), Rieser and Lemon (2008)). In addition we will model the number of "hits" or search results returned by the system after every user turn – this will be a number from 0 to 100.

### 4.2 Action set

See figure 1 for the hierarchical structure of the actions available to the system, representing the combined dialogue management (DM) and NLG task as an inter-related planning problem.

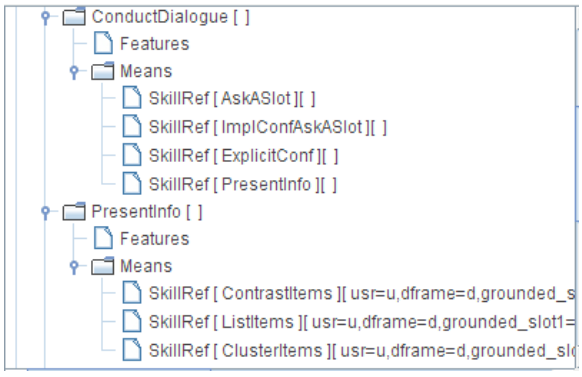


Figure 1: A Hierarchical Plan for NLG and DM

Here we see a top-level "skill" (ConductDialogue) responsible for dialogue management choices in the MDP, and a second level skill (PresentInfo) which is reponsible for the NLG choices. ConductDialogue governs the standard dialogue management options, and decomposes into the 4 possible action choices (or "Means") AskASlot, ImplicitConfirm_and_AskASlot, ExplicitConfirm, and PresentInfo. This allows the system to choose between these 4 types of dialogue act at any time. More interestingly, for the NLG component of the system we have implemented possible 3 action choices (or "Means") for information presentation under PresentInfo: ContrastItems, ListItems, and ClusterItems. ListItems is just the standard list content structuring operator, while ContrastItems and ClusterItems are actions which structure the items presented to the users by contrasting them and clustering them respectively, as shown above.

### 4.3 Reward function

Now that we have our states and actions, we need to define a Reward signal (or "Objective function") for the learning system. This directs the learner in terms of its overall goals (e.g. short dialogues where users rate information presentation highly), while it is up to the learner to find an action policy which meets these goals. What makes the learning problem interesting is that these goals contain conflicting "trade-offs" that the system must learn to balance, based on the state that it is in. For example, the goal to have short dialogues conflicts with the goal to get reliable (i.e. confirmed) search constraints from the user and to present small numbers of items. The learning problem here is then for the system to decide at each turn whether to ask for more information/constraints, confirm (explicitly or implicitly) the exisiting information/constraints, or to List, Contrast, or Cluster the current items returned from the DB. Note that the system can decide to immediately present information in some way to the user even if not all slots are filled or confirmed. This leaves open the option for the system to exploit a "good" information presentation situation (such as having only 2 database items to tell the user about via a Contrast) even if the DM situation (e.g. having only 2 filled slots) is not in itself very rewarding. In this way the NLG and DM decisions are jointly optimised in this setup.

For training a system to be deployed with real

users, this reward/objective function would be developed based on a PARADISE-style (Walker et al., 2000) analysis of a small amount of Wizard-of-Oz data (Walker et al., 1998; Rieser and Lemon, 2008). Here, to prove the concept of statistical planning for NLG, we simply show that the learner can jointly optimize the NLG and dialogue management decisions based on a complex reward signal. Nothing depends on the particular values chosen here – they are for illustration only and can be estimated from suitable data.

The overall reward for each dialogue conducted by the system has 3 components: *completion reward, turn penalty*, and *presentation reward/penalty*. Turn penalty is simply -1 per system turn. The completion reward is the % probability that the items presented to the user correctly meet their actual search constraints, and is therefore a function of the number of filled or confirmed slots. For example, if all 3 slots are confirmed, then (in this noise model) we have 100% chance of having the search constraints correct. The number of filled/confirmed slots stochastically determines the number of items that the system can present to the user if it decides to enter the presentation phase. For example if 3 slots are filled, then 0-10 items will be presented to the user, if 2 slots are filled 0-20 items are retrieved, if 1 slot, 0-100 items. Again, in a real application, these distributions would be estimated from data.

The presentation reward (PR) for each information presentation action is defined as follows, for $i =$ number of items to be presented:

- ListItems: $0 \leq i \leq 3 : PR = 100; 4 \leq i \leq 8 : PR = 0; 9 \leq i : PR = -100$
- ContrastItems: $i \leq 1 : PR = -100; 2 \leq i \leq 6 : PR = 300; 7 \leq i : PR = -100$
- ClusterItems: $0 \leq i \leq 5 : PR = -100; 5 \leq i \leq 8 : PR = 0; 9 \leq i : PR = 300$

This range of rewards/penalties, together with those for filled and confirmed slots and dialogue length provides a complex environment within which the learner must explore different trade-offs.

### 4.4 Environment and User simulation

For training a policy given this definition of states, actions, and rewards, we also need an environment simulation that responds appropriately to system actions. Here the environment is not only the user, but also the database from which items are retrieved for presentation to the user. For policy exploration we use a simple bigram stochastic user simulation with probabilities estimated from COMMUNICATOR data, similar to (Georgila et al., 2006). At each system turn, a number of database hits is randomly determined as a function of the number of filled search constraint slots, as described above. Note that this user simulation does not need to respond directly to the NLG decisions of the system, since the dialogue closes as soon as the system decides to present information (in whatever manner) to the user. A central open question for this type of MDP model of NLG is how to develop "good" user simulations that are sensitive to system NLG choices (Janarthanam and Lemon, 2008).

### 4.5 The "RL-majority" Baseline policy

In contrast to other work on policy learning, which only uses hand-coded systems for comparison, we choose a more challenging baseline. This is because hand-coded policies have been shown to be inferior to learned policies in numerous studies, e.g. (Levin and Pieraccini, 1997; Singh et al., 2002; Lemon and Liu, 2007; Walker et al., 1998), and also, because our task here is a combination of dialogue management and NLG, we do not want the NLG results to be contaminated by an inferior hand-coded dialogue management policy. We therefore choose to compare against a baseline policy learned for the same problem domain, but where the learner uses the *average* most rewarding action for the NLG component (in this case, Cluster items). We call this the "RL-majority" baseline, because it is the RL analogue of a majority class baseline. This baseline policy does not have access to the "DB hits" feature for decisions under PresentInfo (it does have this feature for the top level decisions though), so it learns the average best NLG action rather than attempting to learn the best NLG action for each possible number of DB hits.

### 4.6 Training the policies

We use a hierarchical SARSA Reinforcement Learning algorithm (Sutton and Barto, 1998) with linear function approximation to train the policies. Figure

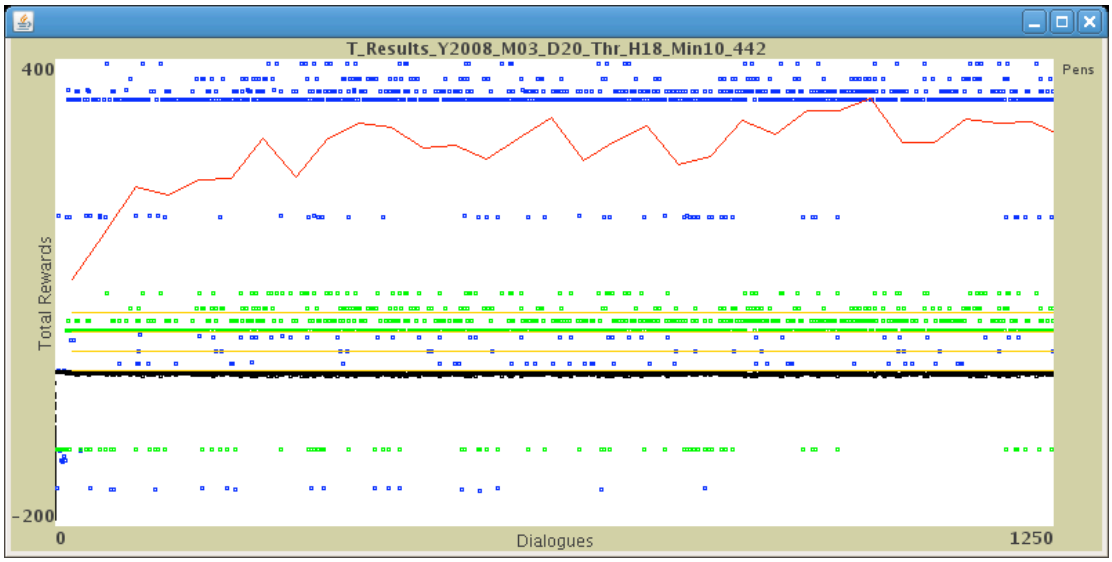


Figure 2: Training the adaptive NLG policy (red line = average dialogue reward)[1]

2 shows learning for the adaptive NLG problem[1].

Here we see that after 1250 training dialogues the system has learned to find a high average reward for the combined NLG and DM problem. At the start of training the system explores bad actions in some states, for example the minimum reward gained in early training is -153, obtained by contrasting more than 7 items (-100) when only 1 slot is filled (-50) after 3 system turns (-3). However, by the end of this training run, the system is able to consistently obtain the best possible rewards given the dialogue situation, for example gaining a top reward of 396 for either Contrast or Cluster of appropriate numbers of items (+300), when all slots are confirmed (+100) in system 4 turns (-4). Where no +300 presentation reward is possible (i.e. $i = 1, 7$, or 8) the system has learned to Cluster or List (when i=1) the items after filling and confirming all slots. A similar graph can be shown for training the Baseline policy.

### 4.7 Testing

We trained both policies multiple times until convergence (approx. 10K cycles), selected the best policy in each case, and tested them (with stochastic simulated users) for 550 test dialogues each. Table 1

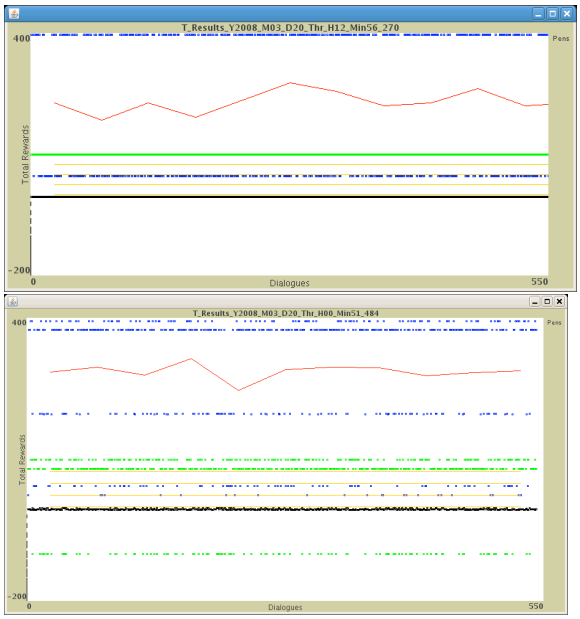


Table 1: Testing the Baseline (top, av. =224.5) and Adaptive (bottom, av. =286.9) NLG policies

[1]In the training/testing graphs red lines show average reward over windows of 50 dialogues, and for each dialogue blue dots show total reward (including NLG reward), black dots show length penalty, and green show completion reward per dialogue.

| Policy | Av. Reward | Av. length |
|---|---|---|
| Baseline Learned | 224.5 | 4.0 |
| Adaptive NLG Learned | 286.9* | 4.98 |

Table 2: Results: learned baseline vs. adaptive NLG policies. ($* = p < 0.001$)

shows the performance of the 2 policies during testing (top= baseline NLG, bottom = adaptive NLG), and the results are presented in table 2.

These results demonstrate a relative increase in reward of 27.8% for the adaptive NLG system. The adaptive NLG system has learned fine-grained local trade-offs for its NLG decisions, which are not available to the baseline system.

So what has been learned? Here is an example dialogue with the adaptive NLG system:
`System`: How can I help you? `(greet)`
`User:` I want a cheap chinese restaurant. (2 slots filled, 2 database items returned)
`System:` Ok. The Golden Wok is cheap and central, and the Noodle bar is cheap but in the south of the city `(Contrast)`

Here we can see that the adaptive NLG policy can decide to present information when it is particularly advantageous, even when the information gathering part of the system is not complete. The Baseline learns a similar policy, but is not sensitive to DB hits when choosing *how* to present the information.

## 5 Summary and Future Directions

This paper demonstrates a new data-driven method where the NLG components of dialogue systems can be automatically trained and globally optimized before deployment.

We surveyed standard approaches to NLG, and described general advantages offered by statistical planning models together with solution methods such as Reinforcement Learning. We gave a brief description of MDP models. In section 4 we cast a standard NLG problem as an MDP, defining the state space, action set, and reward function. We saw how Reinforcement Learning can be used to solve this NLG problem at the same time as optimizing dialogue management. We then evaluated the adaptive NLG policy versus a learned RL-majority baseline. The results showed a significant relative increase in reward of 27.8% for the adaptive NLG system. When given a reward signal that provides feedback on content structuring choices (List, Contrast, Cluster) the system learns to avoid bad decisions (e.g. listing lots of items, clustering small numbers of items, contrasting too few or too many items) and to choose the best NLG option available depending on the number of database items returned by the system at any time.

This demonstrates that our approach brings a number of theoretical and practical benefits such as fine-grained adaptation, and automatic optimization. Future challenges include modelling the hierarchical structure of NLG problems using additional hierarchical MDPs, and modelling complex effects of NLG choices on dialogue context using larger feature sets.

Many other NLG decisions could be approached in this way. By using MDPs to represent other NLG problems we can move to a situation where determination of the best lexical items and referring expressions to use in a system utterance, as well as the best syntactic structure and intonation pattern, are all determined by learned strategies, developed by reward-driven learning based on real data. Reinforcement Learning could also be applied to decisions of when and how to use anaphora and ellipsis.

Overall, this leads us to propose a new development cycle for NLG, whereby the more adaptive NLG components of new dialogue systems can be automatically trained and optimized before deployment, and can then be allowed to adapt online to user feedback (through continued monitoring of rewards). Moreover, due to the use of state generalization techniques such as function approximation, NLG will even be possible in previously unseen and unplanned-for situations.

An open question for this type of model is how to develop "good" user simulations that are sensitive to system NLG choices (Janarthanam and Lemon, 2008). Another important topic is how the classifier-based learning techniques of "trainable" NLG (Barzilay and Lapata, 2005; Duboue and McKeown, 2003; Stent et al., 2004; Walker et al., 2007) can be integrated with the MDP approach proposed here. Other avenues to explore are how interactive alignment (Garrod and Pickering, 2001) and semantic coordination in dialogue (Larsson, 2007)

can be modelled in this framework.

## Acknowledgments

The research leading to these results has received funding from the EPSRC (project no. EP/E019501/1) and from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement no. 216594 (CLASSiC project www.classic-project.org)

## References

Regina Barzilay and Mirella Lapata. 2005. Collective content selection for concept-to-text generation. In *Proceedings of EMNLP*.

Giuseppe Carenini and Johanna D. Moore. 2006. Generating and evaluating evaluative arguments. *Artificial Intelligence*, 170(11):925–952.

Vera Demberg and Johanna D. Moore. 2006. Information presentation in spoken dialogue systems. In *Proceedings of EACL*.

Pablo A. Duboue and Kathleen R. McKeown. 2003. Statistical acquisition of content selection rules for natural language generation. In *Proceedings of EMNLP*.

Simon Garrod and Martin Pickering. 2001. Toward a mechanistic psychology of dialogue: The interactive alignment model. In *Proceedings of BI-dialog*.

Kallirroi Georgila, James Henderson, and Oliver Lemon. 2006. User simulation for spoken dialogue systems: Learning and evaluation. In *Proceedings of Interspeech/ICSLP*, pages 1065–1068.

Amy Isard, Jon Oberlander, Ion Androutsopoulos, and Colin Matheson. 2003. Speaking the users' languages. *IEEE Intelligent Systems Magazine*, 18(1):40–45.

Srinivasan Janarthanam and Oliver Lemon. 2008. User simulations for online adaptation and knowledge-alignment in Troubleshooting dialogue systems. In *Proceedings of SEMdial*.

Alexander Koller and Matthew Stone. 2007. Sentence generation as planning. In *Proceedings of ACL*.

Staffan Larsson. 2007. Coordinating on ad-hoc semantic systems in dialogue. In *Proceedings of DECALOG*.

Oliver Lemon and Xingkun Liu. 2007. Dialogue policy learning for combinations of noise and user simulation: transfer results. In *SIGdial*.

E. Levin and R. Pieraccini. 1997. A stochastic model of computer-human interaction for learning dialogue strategies. In *Proceedings of Eurospeech*.

Johanna Moore, Mary Ellen Foster, Oliver Lemon, and Michael White. 2004. Generating tailored, comparative descriptions in spoken dialogue. In *Proc. FLAIRS*.

Alice Oh and Alexander Rudnicky. 2002. Stochastic natural language generation for spoken dialog systems. *Computer, Speech & Language*, 16(3/4):387–407.

Ehud Reiter and Robert Dale. 2000. *Building Natural Language Generation Systems*. CUP.

Verena Rieser and Oliver Lemon. 2008. Learning Effective Multimodal Dialogue Strategies from Wizard-of-Oz data: Bootstrapping and Evaluation. In *Proceedings of ACL*, page (to appear).

Satinder Singh, Diane Litman, Michael Kearns, and Marilyn Walker. 2002. Optimizing dialogue management with reinforcement learning: Experiments with the NJFun system. *Journal of Artificial Intelligence Research (JAIR)*.

Amanda Stent, Rashmi Prasad, and Marilyn Walker. 2004. Trainable sentence planning for complex information presentation in spoken dialog systems. In *Association for Computational Linguistics*.

Matthew Stone, Christine Doran, Bonnie Webber, Tonia Bleam, and Martha Palmer. 2003. Microplanning with communicative intentions: the SPUD system. *Computational Intelligence*, 19(4):311–381.

Richard Sutton and Andrew Barto. 1998. *Reinforcement Learning*. MIT Press.

Marilyn A. Walker, Jeanne C. Fromer, and Shrikanth Narayanan. 1998. Learning optimal dialogue strategies: a case study of a spoken dialogue agent for email. In *Proceedings of ACL*.

Marilyn A. Walker, Candace A. Kamm, and Diane J. Litman. 2000. Towards Developing General Models of Usability with PARADISE. *Natural Language Engineering*, 6(3).

M. Walker, O. Rambow, and M. Rogati. 2001. Spot: A trainable sentence planner. In *In Proc. of the NAACL,*.

Marilyn Walker, S. Whittaker, A. Stent, P. Maloor, J. Moore, M. Johnston, and G. Vasireddy. 2004. User tailored generation in the match multimodal dialogue system. *Cognitive Science*, 28:811–840.

Marilyn Walker, Amanda Stent, François Mairesse, and Rashmi Prasad. 2007. Individual and domain adaptation in sentence planning for dialogue. *Journal of Artificial Intelligence Research (JAIR)*, 30:413–456.

Steve Young. 2000. Probabilistic methods in spoken dialogue systems. *Philosophical Transactions of the Royal Society (Series A)*, 358(1769):1389–1402.