A Constraint Satisfaction Framework for Managing Mixed-Initiative Discourse

Toby Donaldson and Robin Cohen

tjdonald@neumann.uwaterloo.ca, rcohen@watdragon.uwaterloo.ca
Department of Computer Science
University of Waterloo
Waterloo, Ontario, Canada
N2L 3G1

Abstract

We outline a three-part goal-oriented model of turn-taking, and apply it to the design of a constraint-based conversational agent. We give a detailed example of how an agent can use constraints to manage and choose from multiple turn-taking goals available for it to pursue. Our model is flexible enough to allow arbitary turn-taking behaviour on the part of any conversant, and so allows for mixed-initiative interaction between systems and users.

Introduction

In our research, we have developed a model for turn-taking in discourse, for application to the design of intelligent interfaces and general "conversational agents". Our model is goal-oriented and explicitly addresses three main concerns: *why* to take a turn (motivation), *how* to take a turn (which goal to address in the dialogue), and *when* to take a turn (at relevant points in the dialogue). We present an overview of this model below and focus on discussing the following issues raised in the symposium's call for papers:

- characterizing initiative in terms of decisions to dispense with one of several possible dialogue goals, at fortuitous points in the conversation;
- using turn-taking goals, a time sensitive variant of persistent goals, as a knowledge representation scheme for modelling intelligent interaction;
- using a constraint satisfaction (CSP) framework as the basis for algorithms to manage these turn-taking goals.

In this paper, we focus our discussion on how CSPs can be used to manage mixed-initiative interaction, showing a particular example of how a constraint satisfaction framework can be used to allow a conversational agent to choose among multiple turn-taking goals. An agent can take the initiative depending upon what goals it decides to pursue. Initiative influences turn-taking since the conversant with the initiative may want to carefully control who has the floor in order to keep the conversation moving in the direction she wants.

A Three-Step Model for Taking Turns

One of the basic facts of conversation is that participants (try to) take turns speaking. Occasionally, conversants do speak at the same time (this is called *doubletalk*), but this occurs surprisingly infrequently in human-human conversation — usually one person will take control of the conversation (Oreström 1983). We believe that if human-computer conversation is to ever be as dynamic and flexible as human-human conversation, the computer must have an understanding of how to take turns in order to participate in an effective way (e.g. avoid irritating the user). We want to avoid systems that ask too many unnecessary questions ("tag-along systems"), get locked into a certain mode of conversation (e.g allowing the user to only answer with yes/no responses), or interrupt too frequently or at inappropriate times.

We are developing a model of dynamic turn-taking to address these, and other, concerns. Our turn-taking model consists of three main parts: a motivation to take a turn, the adoption of a goal to take a turn, and, finally, the actual execution of a turn-taking goal. We will briefly discuss each of these parts; Figure 1 gives a general overview of our theory.

Motivation: Why Should I Take a Turn?

There are many different reasons why a conversational agent might want to take a turn speaking. For example:

- You are having a conversation with someone, and, to avoid an uncomfortable pause or to simply continue the conversation, you need to say something. In such cases, small-talk replies (such as "How about them Democrats?") can be adequate since the point is just to prevent the conversation from ending, not to make any profound utterance. A related reason is that you sometimes speak simply because you are responding to someone else; for example, if you have just been asked a question, then, even if

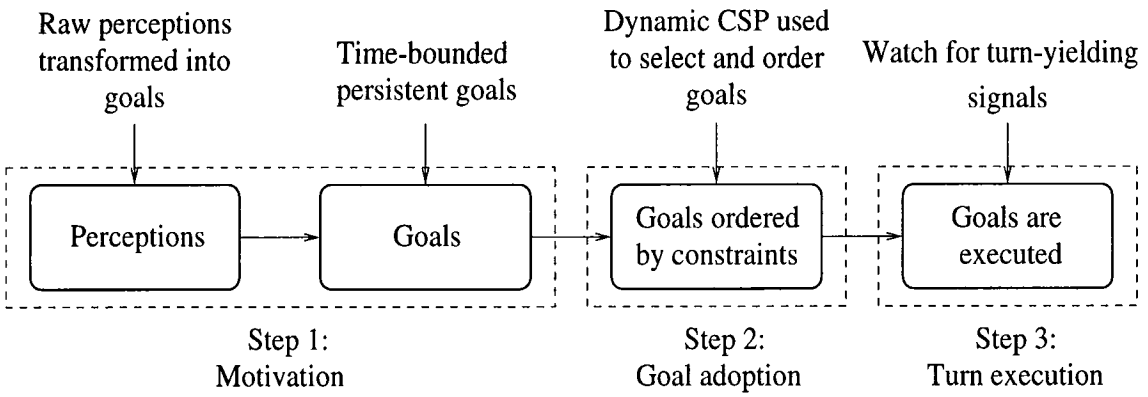

Figure 1: The 3-step model for taking a turn.

you are not sure of the answer, you ought to respond to the question, by default;

- Something the other conversant has said is somehow "physically" hard to understand: you might not have heard an important part of what they said (and so ask them to speak again);
- You are reacting to a typical conversational setting, such as buying a ticket for a concert where the conversants follow a standard turn-taking pattern.

There are also turn-taking motivations which arise when a conversational agent has the overall task of informing or advising the user, such as:

- Another conversant's utterance might cause you to believe that a contradiction has arisen — either you believe some inconsistency, or you believe they believe some inconsistency. One way to resolve an inconsistency is to take a turn and ask the other conversant about it;
- You lack some important piece of information, and so ask the other person (this is the idea behind the *missing axiom theory* of conversation in (Smith, Hipp, & Biermann 1995));
- You have some important piece of information that you want to communicate, e.g. a fire in the building, that someone has just won the lottery, that their mother just called, etc.;

A number of these motivations follow from the simple idea that asking someone else is often the best way to get things done.

Motivations are not sufficient for modelling turn-taking because simply wanting to say something does not mean you should say something. An agent will typically have more than one thing to potentially say at any one time, so it must be able to decide which one thing it will actually speak about. Furthermore, there is necessarily more than one agent in a conversation, and good turn-taking involves listening carefully to the other conversant to see when she is willing to give up the floor.

Goal Adoption: What Should I do When I Want to Take a Turn?

In order to handle multiple turns, and to allow turn-taking goals to be put off until the right moment, we require that the raw motivations be transformed into *turn-taking goals*. We model turn-taking goals as a kind of *time-bounded persistent goal* (Donaldson & Cohen 1996). Time-bounded persistent goals, which are an extension of *persistent achievement goals* as defined by Cohen and Levesque (Cohen & Levesque 1990), take the following general form:

Bounded-persistent-goal(ϕ,T)
While: *simple-goal*(ϕ)
Adopt-when: $B(holds(B\neg\phi, \textit{some-head-of}(T)))$
Drop-when: $B(holds(B\phi, \textit{some-tail-of}(T)))$
$B(holds(B\neg\phi, \textit{some-tail-of}(T)))$
$B(after(T, now))$

This means that the agent has persistent goal to make ϕ true for some interval of time that ends T. Bp means that the agent believes proposition p to be true, and the notation *some-head-of*(T) and *some-tail-of*(T) is shorthand for unspecified intervals of a general time interval T that touch one end of T. A goal is *adopted* when all of the conditions on the While list and Adopt-when list hold, and none of the conditions on the Drop-when list hold. Time-bounded persistent goals are goals that an agent persists in trying to achieve for some finite interval of time; the agent may or may not be certain of the exact end-points of the interval, but it is always aware of its finiteness. Time-bounded persistent goals provide a convenient way to represent "intervals of relevance" in a conversation, and, through the use of explicit adopt/drop conditions, they are very similar to STRIPS planning operators, allowing new

kinds of goals to be conveniently defined.[1]

One of our major concerns is how to deal with multiple turn-taking goals. For example, if someone says to you "Are you hungry? Want to go for lunch?", a number of possible responses might occur to you, depending on the situation: you might need to decide between giving a straight yes/no answer, or expressing your surprise that it is already time for lunch. At any given time, a conversational agent can have more than one turn-taking goal to choose from, and thus must settle upon which goal to actually try to achieve. We have chosen a *dynamic constraint satisfaction* (DCSP) framework in which to manage multiple goals, which we discuss in the section on goal management.

Execution: When Should I Take a Turn?

Finally, once a conversational agent has decided which turn-taking goal to try to execute, it still must take into account one extremely important factor: the other conversant! Since avoiding doubletalk is important, an agent with a goal to take a turn must be careful to take its turn at the right time in the conversation. To do this in human-human conversation, a person must be a good listener: she must look for the appropriate turn-yielding signals from the current speaker, signals such as pauses, changes in inflection, grammatical boundaries, gestures (Oreström 1983; Goldman-Eisler 1968). For example, a pause is often a good indicator of a speaker's willingness to give up the floor, but not always. Sometimes a pause simply indicates that the speaker is thinking, and so perhaps does not want to be interrupted. When a number of *turn-yielding* signals occur at the same time, that likely indicates a good time for a person to take a turn. For human-computer conversation, we would ultimately like to develop a system that could take advantage of these same signals in order to engage in smooth conversational turn-taking.

Tension exists between the desire for an agent to execute its turn-taking goals, and the desire not to haphazardly interrupt the other speaker. Sometimes, the agent may guess wrong (i.e. not interrupt when it should have interrupted), and, due to the time-bounded nature of turn-taking goals, be forced to drop a goal because the time of relevance has passed. The opposite problem is jumping in at the wrong time, which causes the conversation to flow less smoothly.

Summary

This section has outlined in full our proposed model for turn-taking in discourse. These three steps are meant to be occurring continuously and in parallel throughout the conversation. Motivations can be occurring all the time, a conversational agent can almost always be thinking, and an agent should be ready to speak at almost any time, according to the actions of the other conversant. In the remainder of the paper, we focus on how a conversational agent can use a constraint satisfaction framework to manage the various turn-taking goals that can arise in a conversation. We also use the sample application of advice-giving, where the conversational agent is an advising system, making decisions in order to interact with a user.

[1] One potentially interesting consequence of this kind of flexibility is that it should be possible to change the level of persistence for these goals, and so provide some empirical support for Cohen and Levesque's theory of persistence. For example, we could make agents less persistent by requiring less strict drop conditions, and show that this results in less appropriate turn-taking behaviour.

Managing Goals via Dynamic Constraint Satisfaction

Constraint satisfaction problems (CSPs) are a popular formalism for representing problems wherein the goal is to find a consistent assignment of values to variables (Tsang 1993). Various constraints hold between variables, and the problem is to find an assignment that breaks as few constraints as possible (a CSP can be over-constrained, meaning that every possible assignment of values to variables breaks at least one constraint). We propose to manage the turn-taking goals that arise in multi-utterance discourse by using a CSP to manage the selection of turn-taking goals. In our model, every utterance a speaker makes will cause the listener to adopt a number of potential turn-taking goals; the listener must then order these goals, and choose the most appropriate one to actually pursue.

Turn-taking CSPs

A CSP consists of variables, possible domain values for each variable, and constraints between variables. In our domain, variables act as place-holders for goals (they can be thought of as, roughly, being the working memory of a conversational agent); we refer to the resulting CSP as a *turn-taking CSP*. A conversational agent's turn-taking CSP will have k variables $V_1, \ldots V_k$, each capable of holding one turn-taking goal. If an agent is required to act immediately, then it will first try to achieve the goal stored in V_1, then V_2, and so on. The goals that fill the variables, $G_1, \ldots, G_n$, are triggered by utterances from the user, and stored in one large pool of goals. The CSP must decide which goals should go into the V_i's, and in what order. It does this by examining the constraints that hold between the variables, and trying to assign goals that break as few constraints as possible.

Given that an agent could be required to speak at any time (i.e. the choice of when to speak in a conversation often depends on when the current speaker is willing to give up her turn), and given that there is no guarantee that the relevant constraints will be not be so constraining as to make a perfect solution impossible, we propose to solve turn-taking CSPs using *heuristic repair* methods, algorithms similar in style to the *min-conflicts* heuristic (Minton *et al.* 1990)

and GSAT algorithms (Selman, Kautz, & Cohen 1994). Such algorithms start from a completely instantiated CSP, choose a variable according to some heuristic, and then choose a value for that variable by another heuristic (these heuristics often use some kind of randomness). This is a kind of hill-climbing search that has been shown to be highly effective in certain domains. This method is appropriate for solving turn-taking CSPs since it provides an anytime algorithm, it handles over-constrained and under-constrained problems in the same manner, it allows for the re-use of previous solutions, it is memory efficient, and it can sometimes be very time-efficient.[2]

It is our claim that the kind of CSP developed here, which is appropriate for handling single utterances, can be extended to handle multiple utterance discourse in a straightforward way. Note that a single turn can easily consist of multiple utterances, thus our model allows for the possibility of intra-turn interruptions. Below we give an example of how a single utterance triggers various goals in the system, and how the listener orders these goals via constraint satisfaction. When new utterances appear, this cycle repeats; the turn-taking CSP changes since, at least, old goals disappear and new goals arrive. Consideration of multiple utterances will require attention to other linguistic concepts, such as focus, and the use of cue phrases. Thus, multi-utterance discourse is a problem that must be modelled as a *dynamic* CSP (e.g. (Dechter & Dechter 1988; Mittal & Falkenhainer 1990)). We comment briefly on how to approach the problem of dynamic CSPs in the concluding section of the paper.

Kinds of Goals

A turn-taking CSP orders turn-taking goals. We use a very simple notion of goal to model turn-taking goals:

- Each turn-taking goal is associated with exactly one action;
- Turn-taking goals are assumed to be *persistent*, in the sense that an agent will not adopt/drop them in a highly erratic manner;
- Each turn-taking goal will be of a certain type (given below);
- Turn-taking goals are associated with a temporal interval of relevance, which is meant to help model important ideas such as focus, which will become important when multiple utterances and turns come into play.

The particular types of turn-taking goals that are relevant depend upon the type of discourse in which one is engaging. For advice-giving settings, we have identified the following turn-taking goal-types:

- repair goals:
  - information repair goals (i.e. fixing of faulty information that a conversant is believed to possess);
  - contradiction repair (i.e. reconciliation of contradictory utterances/beliefs);
- clarification goals:
  - domain plan clarification (i.e. determining the higher-level goals behind the utterance);
  - meaning clarification (e.g. requesting the meaning of an unfamiliar word used by the speaker);
  - linguistic clarification (i.e. cases where some sort of noise, such as a spelling mistake in written discourse, has prevented the listener from clearly interpreting the utterance);
- information-seeking goals;
- question-answering goals.

This list is not intended to be exhaustive; determining which types of goals are most useful for which domains is a topic for future investigation.

Another factor which we believe will influence the kinds of goals available are the actions that can be associated with the goals. If a system has a domain plan clarification goal, then when it comes time to try to achieve that goal, the system can hand over control of the conversation to a module that handles only that type of subdialogue. When the module has finished executing, it can then return control back to the CSP.[3]

CSP Constraints

The turn-taking CSP orders turn-taking goals by selecting goals that best satisfy the constraints on the CSP variables. In this section, we discuss what kinds of constraints we intend to put on a turn-taking CSP:

General Constraints These are constraints that refer to general properties of turn-taking goals:

- No two variables should hold the very same goal;
- If G_a is a sub-goal of G_b, then G_a should come before G_b;
- If the endpoint of G_a's interval of relevance is before, or coincident with, the beginning point of G_b's interval of relevance, then G_a should come before G_b.

[2] The performance of local search is apparently quite sensitive to the domain, heuristic, and representation used. So far, experimentation is the best way to determine if a local search algorithm will provide better performance than a standard CSP back-tracking style algorithm. We are currently investigating the use of local search algorithms in a number of CSP domains, including turn-taking CSPs.

[3] A particular subdialogue module itself need not explicitly deal with goals in the same way as the turn-taking CSP does (although, of course, it would need to communicate what it has achieved), nor does it need to offer the same flexibility.

Goal-type Constraints These constraints refer to the *type* of a goal, and we refer here only to the goal types listed previously. Again, these constraints are the initial ones we would like to consider, and we expect they may well change and be made more precise through experience:

- related clarification and repair goals come before related information-seeking goals (rationale: information obtained while a potential problem is pending may be bad information that causes problems later);
- related linguistic clarification goals come before related meaning clarification goals (rationale: before you can understand the meaning of the linguistic utterance, you should make sure that you have heard the utterance clearly);
- related meaning clarification goals come before related domain plan recognition clarification goals (rationale: you should make sure you understand an utterance before you start using it to infer a higher-level plan);
- related clarification goals should be done before related answering goals (rationale: giving an answer while an ambiguity exists could lead to giving an unhelpful or even incorrect answer).

Example: Ordering Goals in Response to a Single Utterance

As a simple example of how a conversational agent can use a turn-taking CSP to go from hearing an utterance to deciding which turn-taking goal to pursue, we use a simple example from the course-advising domain. We assume a student is conversing with a system acting as a course advisor; the student can ask questions about courses, their pre-requisites, program options, etc., and the system possesses the necessary knowledge to provide accurate and helpful responses.

Suppose the student asks *Can I take CS150?*. We take this to be a simple question, which triggers a number of turn-taking goals:

G_1: question-answering (the user has asked a question, so answer it!);

G_2: meaning clarification (find out what semester the student is referring to);

G_3: domain plan clarification (find out what domain plan the user is pursuing);[4]

G_4: information-seeking goal (find out if the student has taken the necessary pre-requisites).

[4] We mean by this that the system should try to find out what higher-level goal the student is trying to achieve by taking the course, since a cooperative response to the user may be different depending upon the possible faults in the student's overall plan. See (Cohen, Schmidt, & van Beek 1994) for more details.

For simplicity, we make a number of assumptions. We assume that these are the only goals that the system needs to consider, and use four CSP variables, $V_1, \ldots, V_4$. We assume that G_1 is a super-goal of the other three goals, all the goals are related, and that no time-interval constraints come into play (i.e. we are only worried about what to say for the very next turn).

We also need to specify a scoring system for sets of constraints. For this example we define a simple scoring function f that adds a penalty amount for every constraint violation an instantiation incurs. For every domain constraint that is broken, f adds a penalty of 5, and for every goal-type constraint that is broken, f adds a penalty of 1. For example, if $A_1 = \{V_1 : G_1, V_2 : G_2, V_3 : G_3, V_4 : G_4\}$ and $A_2 = \{V_1 : G_4, V_2 : G_3, V_3 : G_2, V_4 : G_1\}$, then $f(A_1) = 15$ and $f(A_2) = 4$.[5] The problem is to find an assignment of goals to variables that minimizes f.

As discussed previously, we propose to solve this CSP using heuristic repair methods. We will not trace through a detailed example, but the operation of, for instance, the min-conflicts heuristic is straightforward. First, an initial solution is chosen, either randomly or by some other means. Next, the following procedure is repeated indefinitely:

- select a variable that is participating in a conflict;
- assign to this variable the value that decreases the score the most;

In both steps, ties are broken randomly. This process is repeated indefinitely, until either a perfect solution is found, or some re-start mechanism is called (i.e. to escape from local minima). At any time during this process, the best solution found so far can be returned.

Suppose the system hears the utterance, triggers all the goals, and then decides that it has spent enough time thinking, and must speak.[6] By thinking is meant that it has run some heuristic repair algorithm on its turn-taking CSP. Suppose that the best solution it found was A_2, and that it does no more thinking (a simplifying assumption, since new utterances will trigger new goals that should be considered). It would:

1. (G_4) Try to seek information about whether or not the student has achieved the necessary pre-requisites;
2. (G_3) Try to figure out what the student's domain plan is;

[5] The notation $V_i : G_a$ means that variable V_i is assigned value G_a.

[6] Deciding when to speak is step 3 of our turn-taking model, and we will not address this problem here. Note that (Carletta 1992) has shown that a speaker may need to take a turn, prematurely when the other conversant has unexpectedly given up the floor.

3. (G_2) Try to determine which semester the CS150 course the student is referring to occurs in;
4. (G_1) Try to answer the student's original question.

This is not an optimal ordering, but it is a reasonable set of actions to pursue; given more time to think, the system may have found an ordering with fewer penalty points (e.g. $f(\{V_1 : G_2, V_2 : G_3, V_3 : G_4, V_4 : G_1\}) = 1$). Thus, when the system is required to speak, it would perform the action associated with G_4. After that action is completed, and if no more processing has been done by the system, then it would perform the action associated with G_3, and so on.

Discussion

In this paper, we have studied how an intelligent system interacting with a user or with other intelligent agents can manage a dialogue in which direction and control shift among participants. Our focus has been on specifying how a system can decide when to take a turn and what to say, at any given point in the dialogue.

Some previous work has dealt more or less explicitly with turn-taking and initiative. Traum & Hinkelman (1992) present a model of conversation acts for task-oriented dialogue that includes both a very general description of turn-taking actions, and the idea of initiative. They define the conversant with the *local initiative* as the one with "the most recent discourse obligation", and the initiator as the conversant who performs the *initiate* act to begin a new discourse unit. Walker & Whittaker (1990) use the idea of *control*, which is similar to initiative, and discuss, for example, when it is appropriate for a listener to interrupt in terms of a tracking of the discourse to date. Smith et al (1995) describe a dialog system that uses the *missing axiom theory* as its dialogue control mechanism, which causes an agent to take turn when it lacks an "axiom" to prove the theorems it is concerned with.

Work has also been done on a number of related issues. For example, previous work in advising dialogs (Chu-Carroll & Carberry 1995; Cohen, Schmidt, & van Beek 1994) has dealt with the issue of knowing when and how to initiate a particular *kind* of subdialogue, such as information-seeking, meaning-negotiation, and clarification subdialogues. The work presented here goes further in terms of offering a general framework in which such kinds of dialogues can by dynamically initiated and controlled.

Our work addresses more specifically the topic of mixed-initiative interaction in the following way. Our design is primarily one where a system represents its own dialogue goals and then determines when it is best to take a turn during interaction with a user. But our model also allows for the user to take a turn to do something non-standard in its interaction with the system. For instance, a user may elect to answer a question with a question, rather than an answer. A system which takes its turns according to a collection of dialogue goals can simply add to its list the goal of answering the current question of the user; then the importance of this goal can be judged with respect to the previous goal of advising or clarifying (the reason why the system generated the question to the user in the first place). In this respect, the model which we propose, employing time-bounded persistent goals, allowing any goal which currently persists to be a candidate for the system's next move, and then specifically addressing which goal to select and which point in the interaction to take a turn, is a model which provides for great flexibility in the structure of interaction between system and user.

For future work, we are considering ways to augment local search for solving CSPs. One intriguing combination is the use of a simple case-based retrieval system for initiating and augmenting the heuristic repair search. Everyday experience suggests that many conversational utterances are quick "canned" responses, in answer to common questions (such as what the time is, or where someone's office is, etc.). Instead of solving these turn-taking problems by *searching* for an appropriate ordering of turn-taking goals, it seems more plausible that people are simply *retrieving* a case that they have encountered before, and using that to guide their current actions (i.e. either by executing that particular case, or quickly adapting it for the current situation). Thus, we can augment the heuristic repair method by saving old cases (i.e. the circumstances of a previous conversation, and the goal-ordering that was used in that situation), and then, before beginning the repair, searching the case-base to find the best-matching previous case to initialize the turn-taking CSP. Sometimes, the retrieved case may be a good enough match for the current situation and so can be used without resorting to any search at all (e.g. this could happen with small-talk, as when someone says "How are you today?" and you typically respond with "Not bad. And yourself?"). Note that the use of case-based techniques is not tied to this particular problem domain, but is a general method for augmenting *any* CSP solution method where old solutions are available.

Another direction for future work is to specify more fully how dynamic CSPs can be represented and managed. One can think of a dynamic CSP as a collection of static CSPs (indexed by a time parameter). In general, *any* list of static CSPs could be considered a dynamic CSP, but we think it is reasonable to carefully limit how adjacent CSPs must be related; for example, we might require that each successive static CSP have constraints no tighter than the previous CSPs, or that variables may be added, but never deleted. Using this approach, we are currently investigating a dynamic CSP formalism where a dynamic CSP is defined to consist of an initial static CSP, and a "change" re-

lation that explicitly specifies how the CSP is allowed to change over time.

References

Carletta, J. 1992. *Risk-taking and recovery in task-oriented dialogue.* Ph.D. Dissertation, University of Edinburgh Dept. of Artificial Intelligence.

Chu-Carroll, J., and Carberry, S. 1995. Generating information-sharing subdialogues in expert-user consultation. In *Proceedings of IJCAI '95*, 1243–1250.

Cohen, P., and Levesque, H. 1990. Intention is choice with commitment. *Artificial Intelligence* 42:213–261.

Cohen, R.; Schmidt, K.; and van Beek, P. 1994. A framework for soliciting clarification from users during plan recognition. In *Proceedings of the 4th International Conference on User Modeling.*

Dechter, R., and Dechter, A. 1988. Belief maintenance in dynamic constraint networks. In *Proceedings of AAAI-88*, 37–42.

Donaldson, T., and Cohen, R. 1996. Turn-taking in discourse and its application to intelligent agents. In *AAAI-96 Workshop on Agent Modeling.*

Goldman-Eisler, F. 1968. *Psycholinguistics, Experiments in Spontaneous Speech.* Academic Press.

Minton, S.; Johnston, M.; Philips, A.; and Laird, P. 1990. Solving large-scale constraint satisfaction and scheduling problems using a heuristic repair method. In *Proceedings of AAAI-90*, 17–24.

Mittal, S., and Falkenhainer, B. 1990. Dynamic constraint satisfaction problems. In *Proceedings of AAAI-90.*

Oreström, B. 1983. *Turn-taking In English Conversation.* CWK Gleerup.

Selman, B.; Kautz, H.; and Cohen, B. 1994. Noise strategies for improving local search. In *Proceedings of AAAI-94*, 337–343.

Smith, R.; Hipp, D.; and Biermann, A. 1995. An architecture for voice dialog systems based on prolog-style theorem proving. *Computational Linguistics* 21(3):281–320.

Traum, D., and Hinkelman, E. 1992. Conversation acts in task-oriented spoken dialogue. *Computational Intelligence* 8(3):575–599.

Tsang, E. 1993. *Foundations of Constraint Satisfaction.* Academic Press.

Walker, M., and Whittaker, S. 1990. Mixed initiative in dialogue: An investigation into discourse segmentation. In *Proceedings of the 28th ACL Conference*, 70–78. Pittsburgh.