Language Technology II: Natural Language Dialogue

Dialogue Management

Ivana Kruijff-Korbayová
ivana.kruijff@dfki.de

Outline

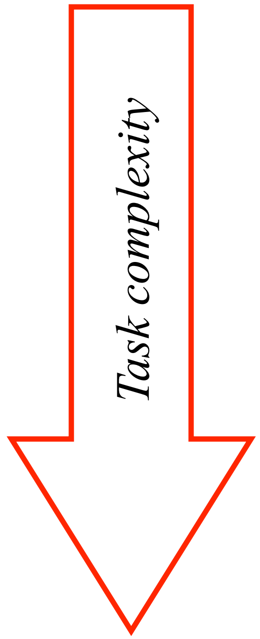
- Tasks of dialogue management
- Finite State-Based DM
- Frame-Based DM
- ISU-Based DM
- Current challenges

Tasks of Dialogue Management

- Dialogue flow control
- Dialogue modeling
  - Dialogue context
  - Dialogue moves
- Dialogue act decision making
- Dialogue phenomena:
  - Error handling
  - Initiative and cooperation
  - Adaptivity
  - …

Methods of DM

- Script-based (state machines)
  - Sequence of pre-defined steps (dialogue script)
- Frame-based (also: form-filling)
  - Set of slots to be filled (task template) and corresponding prompts
- Plan-based
  - Collaborative problem solving



Generic paradigm:

- Information-State Update
  - Declarative rules for updating dialogue context

Script-Based DM

Script-Based DM

- Script describes all possible dialogues
- Typically finite state machine
- Set of states and transitions
  – State determines system utterance
  – User utterance determines transition to next state (deterministic)
- No recursion! (= no nested subdialogues)
- Fixed dialogue script
- OK for system-driven interaction

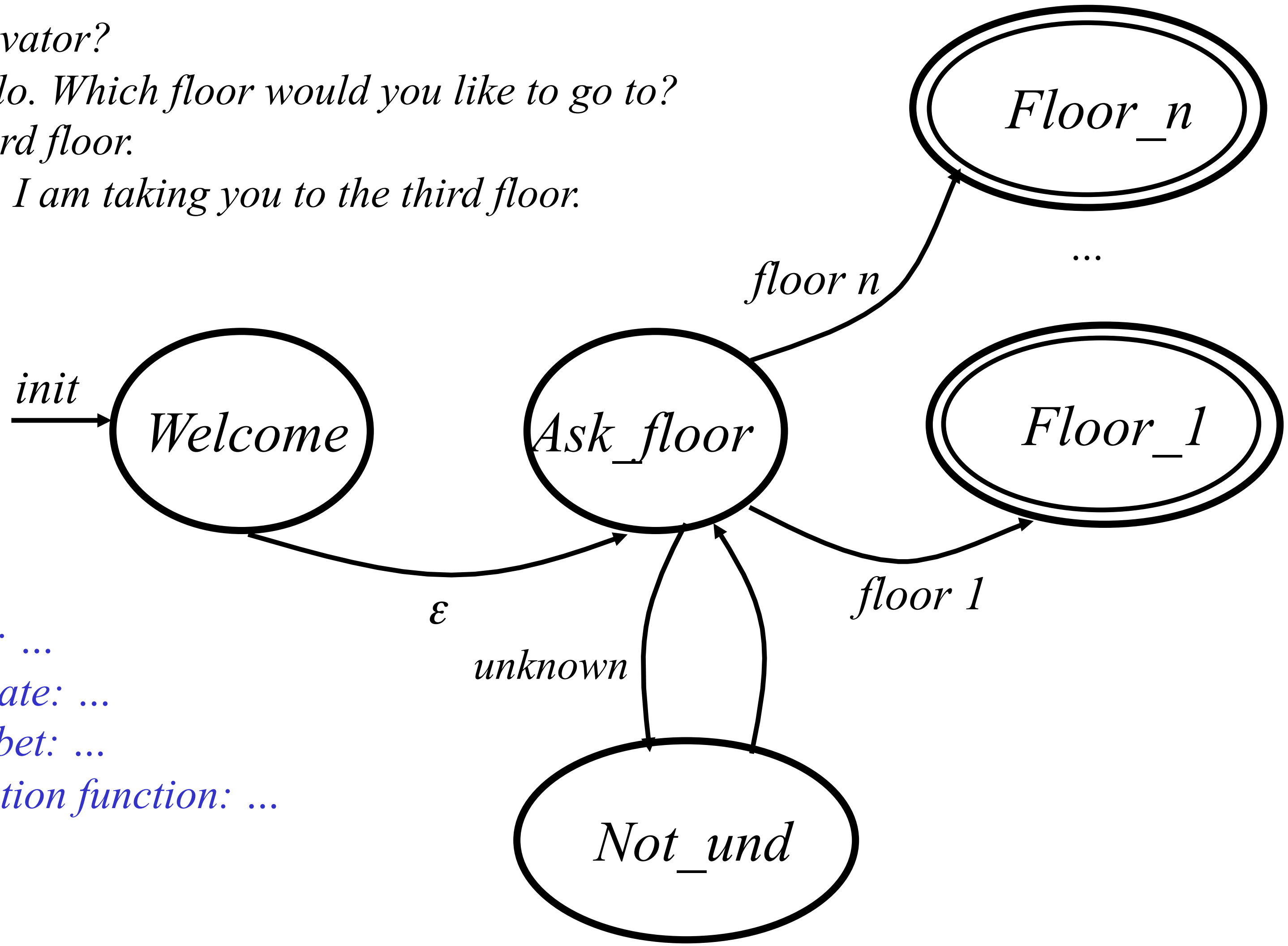
Finite State Machine

- <States, Init-State, Alphabet, Transition-fction>
- Variants: machines having
  – actions associated with states (Moore machine)
  – actions associated with transitions (Mealy machine)
  – multiple start states
  – transitions conditioned on no input symbol (a null)
  – more than one transition for a given symbol and state (nondeterministic finite state machine)
  – states designated as accepting states (recognizer)
  – etc.

See, e.g., NIST http://www.nist.gov/dads/HTML/finiteStateMachine.html

FSM-Based Models

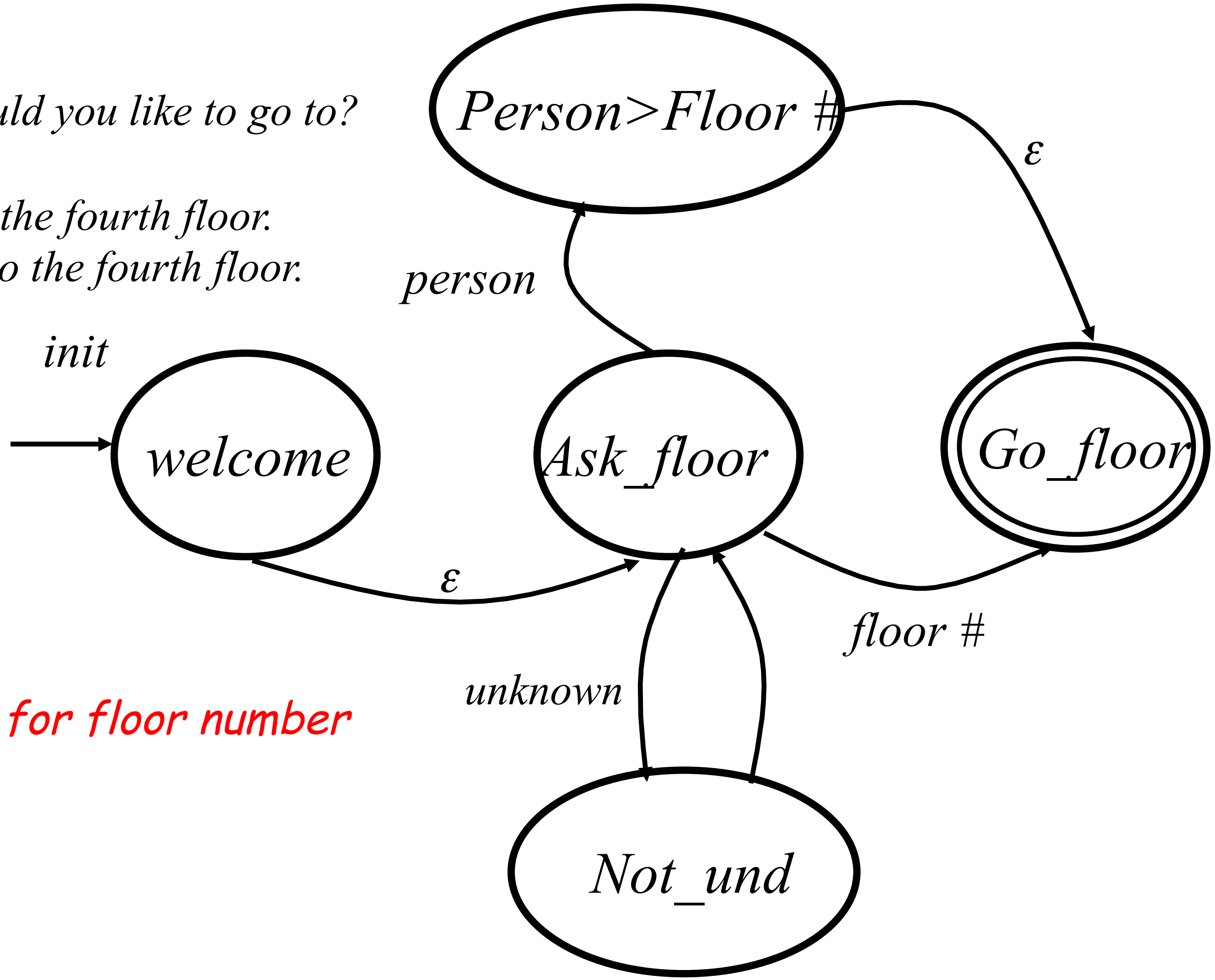
U: Elevator?
S: Hello. Which floor would you like to go to?
U: Third floor.
S: OK, I am taking you to the third floor.



States: ...
Init-State: ...
Alphabet: ...
Transition function: ...

FSM-Based Models

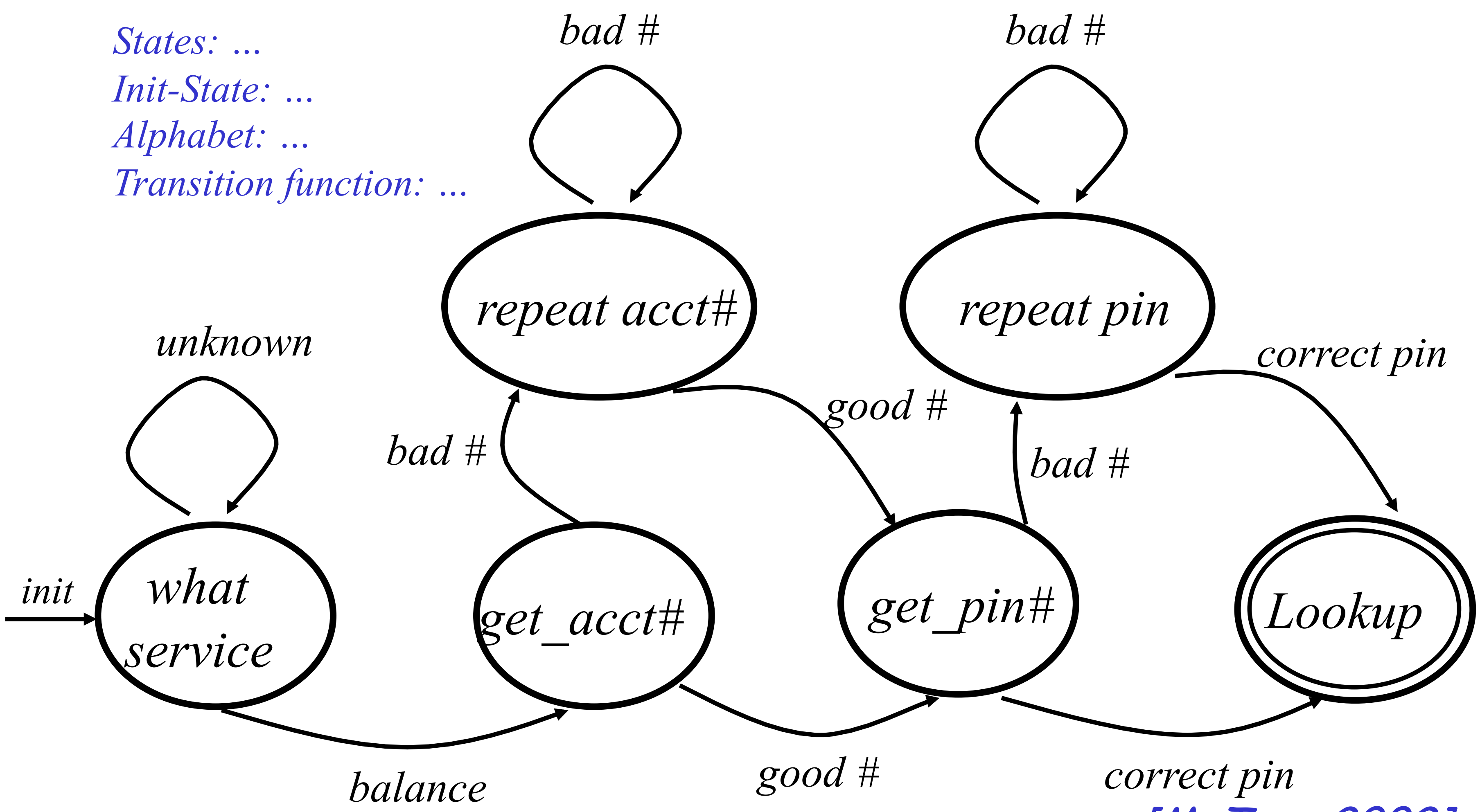
U: Elevator?
S: Hello. Where would you like to go to?
U: Prof. Barry.
S: Prof. Barry is on the fourth floor.
I am taking you to the fourth floor.



Extension: variable for floor number

FSM-Based Models

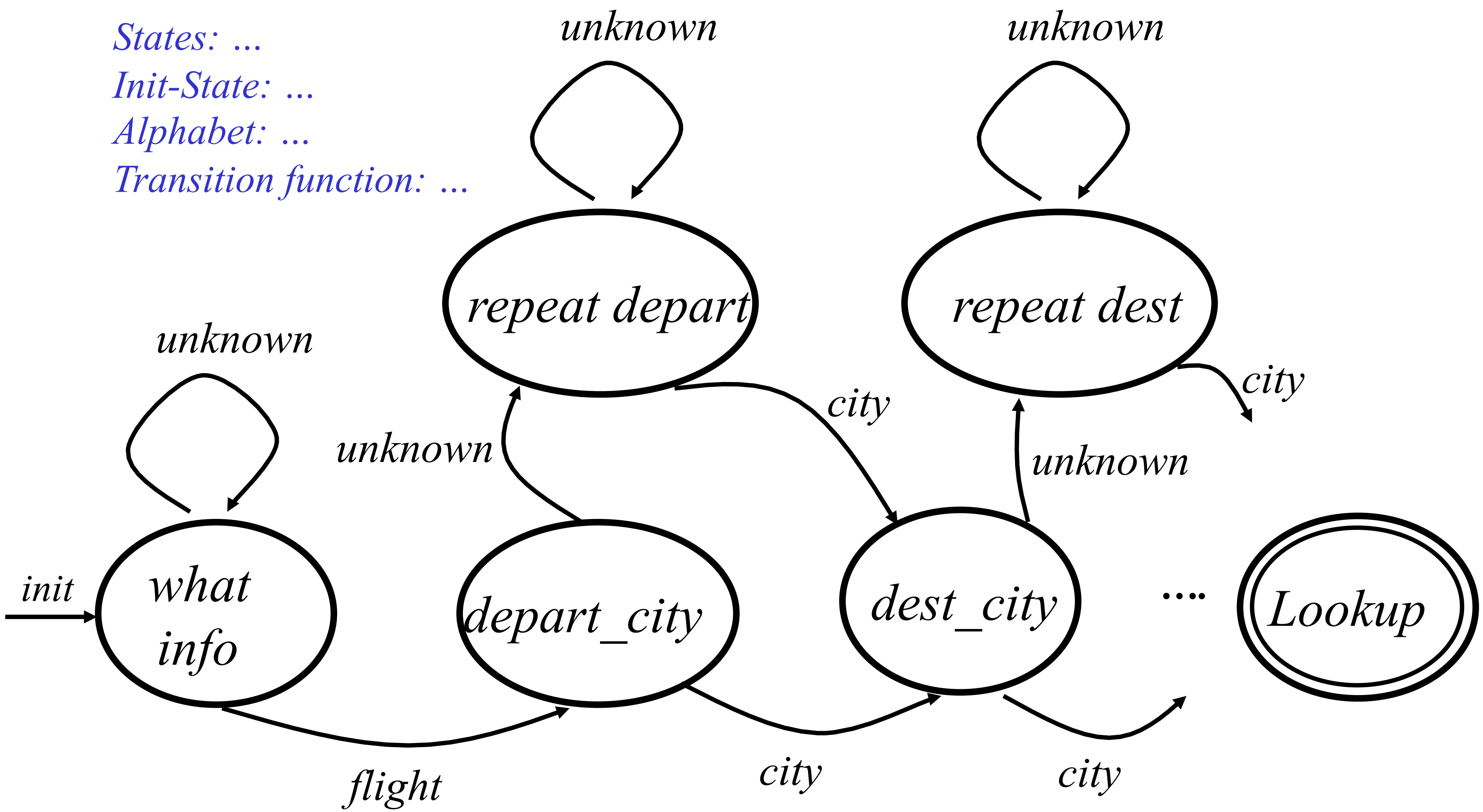
States: ...
Init-State: ...
Alphabet: ...
Transition function: ...



[McTear 2002]

FSM-Based Models

States: ...
Init-State: ...
Alphabet: ...
Transition function: ...



FSM-Based DM: Sum Up

- Advantages
  - Fixed prompts can be pre-recorded
  - Speech recognition and input interpretation can be tuned for each state
- Disadvantages
  - Rigid dialogue flow
  - Inhibiting user initiative
  - Only suitable for simple tasks
  - In principle can make more flexible, but it quickly gets very complex; modular solutions are possible,

Frame-Based DM
(Form Filling)

Frame-Based Models

- Frame (form): what info should be supplied by user

departure_city	?
departure_date	?
destination_city	?
return_date	?
…	

- Dialogue states: which slots are filled
- General routines for what system should do next (given which slots are filled)

Frame-Based Models

S: Where do you want to go?
U: Paris

departure_city	*?*
departure_date	*?*
destination_city	*Paris*
return_date	*?*
…	

S: Where will you travel from?
U: From Berlin.

…

S: When will you travel?
U: August 1st.

departure_city	*Berlin*
departure_date	*1/8/05*
destination_city	*Paris*
return_date	*?*
…	

Frame-Based Models

S: What can I do for you?
U: I want to fly to Paris

departure_city	*?*
departure_date	*?*
destination_city	*Paris*
return_date	*?*
…	

S: Where will you fly from?
U: From Berlin on August 1st.

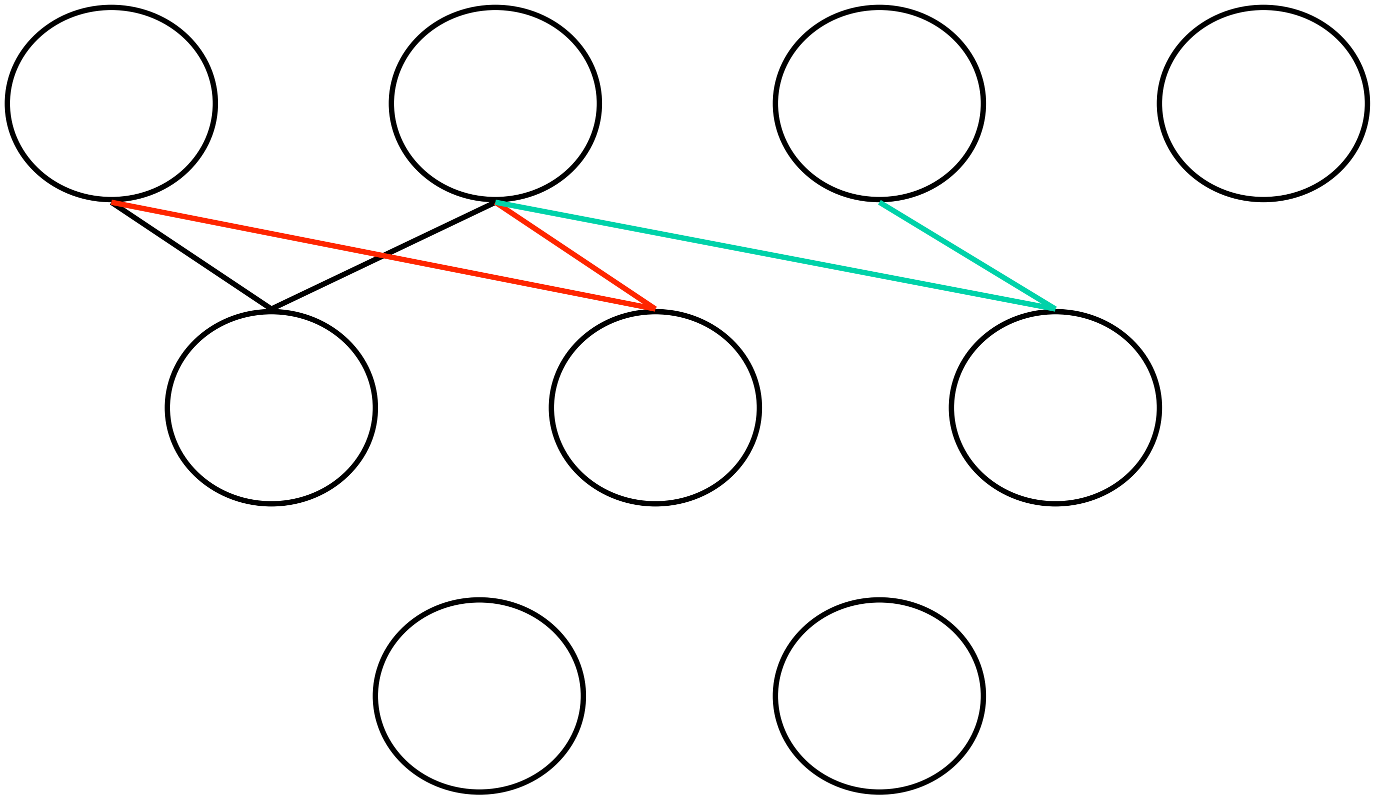
"Overanswering"

departure_city	*Berlin*
departure_date	*1/8/05*
destination_city	*Paris*
return_date	*?*
…	

Frame-Based Models

- Strategies for deciding what to do next
  - Next unfilled slot
  - Slot-combination weighting
  - Ontology-based coherence
- Options for database lookup
  - Delayed (typically; after certain slots filled)
  - Immediate (can be “expensive” = take time, but enables more helpful system behavior)

Slot-Combination Weighting



Ontology-based coherence

1. S: What is the patient's sex?
   U: Female with severe nipple discharge
   S: What is the patient's age?
   U: Fifty five
   S: Is the discharge bilateral?
   U: No

2. S: What is the patient's sex?
   U: Female with severe nipple discharge
   S: Is the discharge bilateral?
   U: No
   S: What is the patient's age?
   U: Fifty five

[Milward&Beveridge 2003]

Problems with Delayed DB Lookup

S: What can I do for you?
U: I want to fly from Berlin to Paris.
S: When would you like to fly?
U: April 1st.
S: At what time would like to fly?
U: In the morning.
S: Sorry. There is no flight from Berlin to Paris on April 1st in the morning.
When would you like to fly?
U: Afternoon.
S: There is one flight. It leaves Berlin at 3 PM and arrives to Paris at 5 PM.

S: What would you like to hear?
U: Play Yesterday.
S: There are 5 songs called Yesterday. Which artist would you like?
U: The Beatles.
S: Sorry, I do not have Yesterday by the Beatles. Would you like another artist?
U: ...

VoiceXML

- Digital document standard for specifying interactive media and voice dialogue
- Used for developing audio and voice response applications
- VoiceXML documents are interpreted by a voice browser
- VoiceXML application collects and processes info, and plays back info

VoiceXML

- Main elements of a VoiceXML document
  - Form: basic unit of functionality
  - Field: prompts for and accepts user input
  - Prompt: sequence of audio elements or TTS messages
  - Audio: audio file or TTS message to play
  - Filled: processes input, can pass control to other forms
- Form Interpretation Algorithm
  - Defines how fields in a form are filled in , and how the fill ordering can be modified
- Global event handlers (e.g., for error handling, help)
  - Define behavior when predefined global conditions occur
- Control transfer conditions and subroutine constructs (= special-purpose programming language)
  - new, more expressive standard: State Chart XML

VoiceXML

See VoiceXML tutorials

http://www.palowireless.com/voicexml/tutorials.asp

e.g.,

http://www.vocomosoft.com/voicexml_tutorial.htm

Or Chapters 1 and 2 of

http://cafe.bevocal.com/docs/tutorial/index.html

give good first steps

Frame-Based DM: Sum Up

- Advantages
  - More flexible dialogue
  - Enables some user initiative
- Disadvantages
  - Speech recognition more difficult, because user input less restricted
  - Not every task can be modeled by a frame

Plan-Based DM

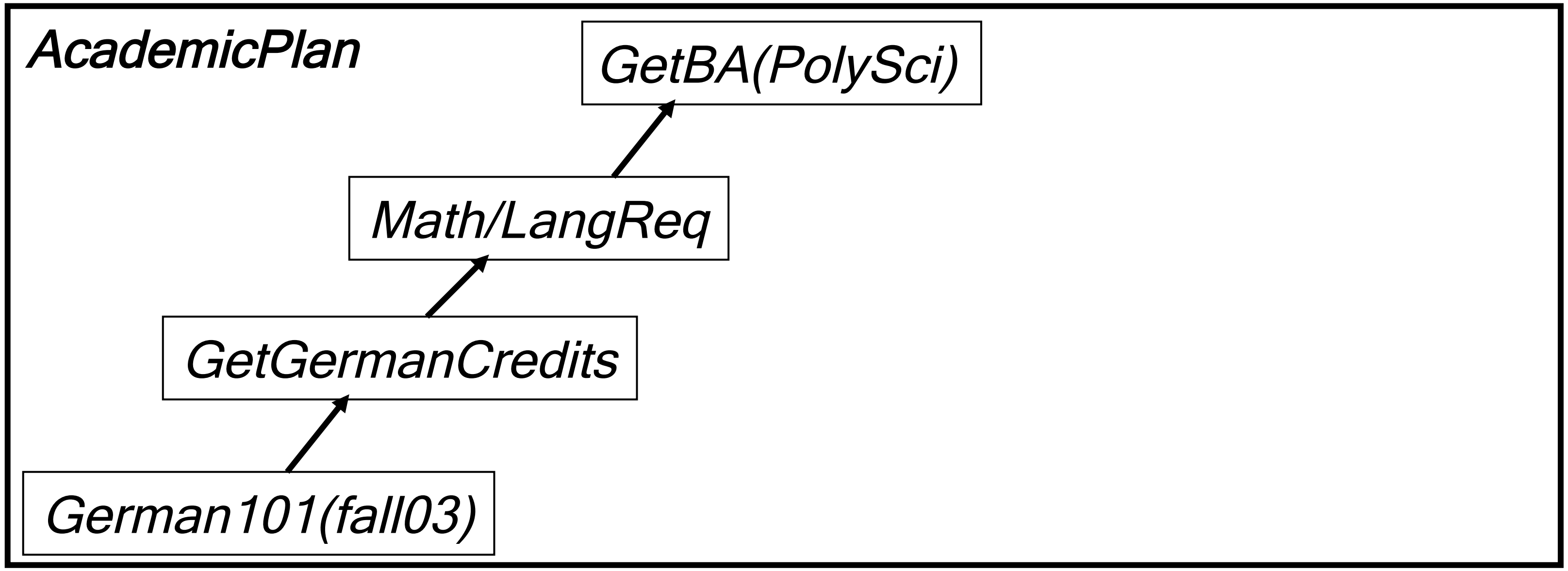
Plan-Based DM

- Communication is a joint activity: Agents communicate to establish common ground, agents collaborate to accomplish a task
- Collaborative problem solving by (rational) agents
  - Neither agent can accomplish the task alone
  - Need joint goals and mutual understanding
  - Agents collaborate to establish and achieve their goals
- Agents have knowledge about solving tasks
  - deciding on goals (objectives): adopt, select, defer, abandon, release
  - Forming plans to achieve goals (recipes)
    - Automated planning: STRIPS; planning operators: actions, preconditions, postconditions
  - Executing plans (acting)
  - Revising decisions (re-planning, abandoning goals, etc.)
- Agents reason about beliefs and actions
- Intention recognition

Intention Recognition

Given: plan for getting a BA

U: I'll take German 101 fall semester.



Collaborative Planning&Acting

User: Send ambulance one to Parma right away.
(initiate (c-adopt (action (send amb1 Parma))))
(initiate (c-select (action (send amb1 Parma))))
System: OK. [sends ambulance]
(complete (c-adopt (action (send amb1 Parma))))
(complete (c-select (action (send amb1 Parma))))
System: Where should we take the victim once we pick them up?
(initiate (c-adopt (resource (hospital ?x))))
User: Rochester General Hospital.
(continue (c-adopt (resource (hospital RocGen))))
System: OK.
(complete (c-adopt (resource (hospital RocGen))))

[Blaylock et al. 2003]

Plan-based DM

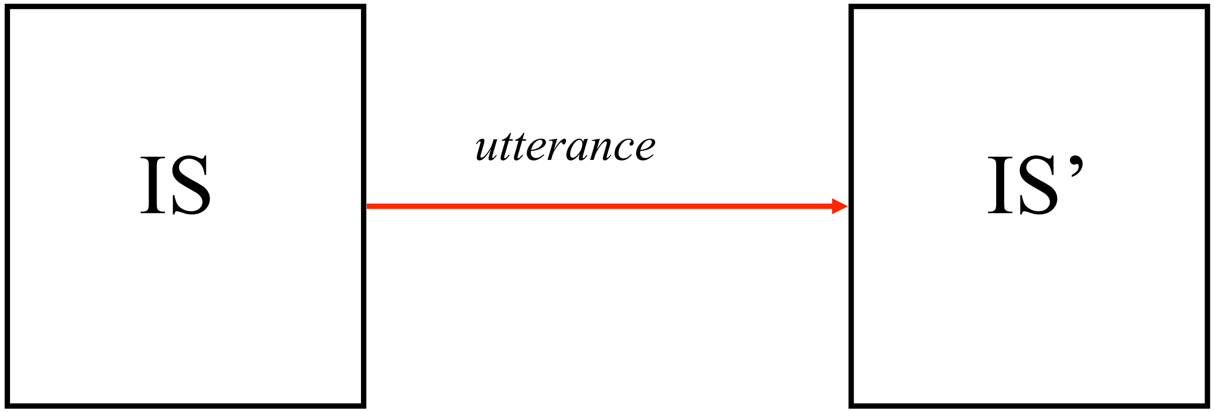
- Advantages
  - Flexibility and adaptivity
  - Any task can be modeled

  … the ultimate solution
- Disadvantages
  - Specifying planning operators is as hard as writing dialogue scripts
  - Intention recognition is a hard problem
  - Lots of reasoning needed

  --> see QUD-based ISU-approach for “shortcuts”

Dialogue Modeling
as
Information State Update

IS Update



Information State

- Representation of the current state of dialogue
- Used by system to
  - Interpret user's turn
  - Decide which external actions to take
  - Decide what to say
  - Store information (dialogue context representation)
- Utterances update information state
- Approaches to DM differ in how IS is represented, what role it plays, what it contains

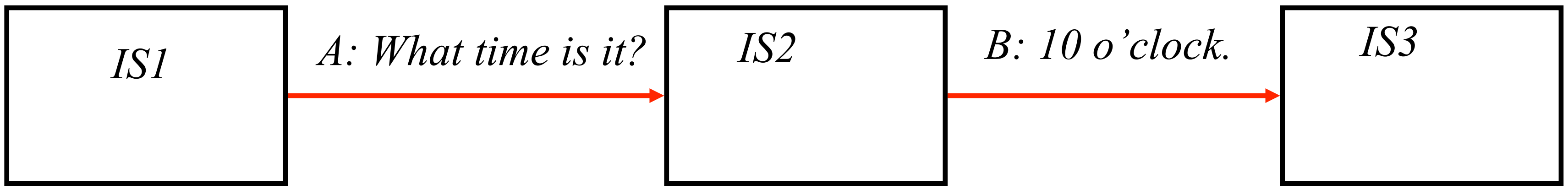
ISU Dialogue Modeling

- Components:
  - a description of the informational components of the IS
    (aspects of common context, participants, common ground, linguistic and intensional structure, commitments, beliefs, intentions, user model...)
  - their formal representation
    (e.g. lists, sets, typed feature structures, DRSs, propositions, modal operators, etc.)
  - set of dialogue moves (DMs) triggering the update of the IS
  - set of update rules governing the IS updates given various conditions of current IS and performed DMs
    (e.g. set of selection rules that license choosing a particular DM to perform given IS)
  - a control strategy to decide which update rule(s) to select at a given point in the dialogue
    (e.g. „pick first that applies", game theory, statistical methods)

IS Update Rules

- Describe possible transitions from one information state to the next

  If <conditions-on-IS-values>
  then <changes-to-IS-values>



Conditions: when a rule is applicable

Effects: how the IS changes

State Machine Model as ISU

- IS: current-state; input
- Update rules:
  - If [state] & [input]
    then [output]; [next-state]

Frame-Based Model as ISU

- IS: task-frame; user's move; system move
- Update rules: e.g.,
  - If [user move = slot X value V] then [fill X with V]
  - If <conditions-on-frame-values>
    then <ask-slot-value Y>

    Decision about next system move is also a rule

ISU-Based Dialogue Modeling

- Task- vs. Dialogue-Structure
  - Task --> dialogue
  - But, dialogue does not have to follow task (execution) structure
- “Dialogue planning”: creating an agenda
  - Task model fills agenda with task-related goals
  - Dialogue manager can add more goals, e.g., for grounding
- Some approaches:
  - QUD-based: Godis (TRINDI, SIRIDUS)
  - Obligation-based: Edis (TRINDI)
  - Agent-based: collaborative problem solving: TALK

QUD-Based ISU Modeling

- Information state in Godis:

$$\begin{bmatrix} \textit{PRIVATE :} & \begin{bmatrix} \textit{AGENDA : stack(Action)} \\ \textit{PLAN : stack(Action)} \\ \textit{BEL : set(Prop)} \end{bmatrix} \\ \textit{SHARED :} & \begin{bmatrix} \textit{COMMITMENTS : set(Prop)} \\ \textit{QUD : stack(Question)} \\ \textit{LU:} \begin{bmatrix} \textit{SPEAKER: Speaker} \\ \textit{MOVES: assocSet(Move)} \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

\+ *module interface variables*

INPUT : String
LATEST-MOVES: Set(Move)
LATEST-SPEAKER: Speaker
NEXT-MOVES: Set(Move)
OUTPUT: String

QUD-Based ISU Modeling

U: "how much does a flight cost?"

- if user asks Q, push respond(Q) on AGENDA
- if respond(Q) on AGENDA and PLAN empty, find plan for Q and load to PLAN
- if findout(Q) first on PLAN, ask Q

S: "where do you want to go?"

U: "Paris"

- if LM=answer(A) and A **about** Q, then add P=Q[A] to SHARED.COM
- if P in SHARED.COM and Q topmost on QUD and P **resolves** Q, then pop QUD
- if P in SHARED.COM and P **fulfils goal** of findout(Q) and findout(Q) on PLAN, then pop PLAN

QUD-Based ISU Modeling

- Sample task plan:

  findout(?x.transport(x))
  findout(?x.dest-city(x))
  findout(?x.depart-city(x))
  findout(?x.dept-month(x))
  findout(?x.dept-day(x))
  findout({?class(economy), ?class(business)}
  consultDB(?x.price(x))
  respond(?x.price(x))

⇒ system's agenda

QUD-Based ISU Modeling

- IS update rule for answer integration: integrateAnswer

$$
\begin{array}{ll}
pre: & \left\{ \begin{array}{l} in(\$SHARED.LU.MOVES,\ answer(A)) \\ fst(\$SHARED.QUD,\ Q) \\ \$DOMAIN{:}about(A,\ Q) \end{array} \right. \\
eff: & \left\{ \begin{array}{l} DOMAIN{:}\ combine(Q,\ A,\ P) \\ add(SHARED.COM,\ P) \end{array} \right.
\end{array}
$$

- Before an answer can be integrated by the system, it must be matched to a question on QUD

QUD-Based ISU Modeling

…

S: ”what class did you have in mind?”

U: ”cheap”

- if consultDB(Q) on PLAN, consult database for answer to Q; store result in PRIVATE.BEL
- if Q on QUD and P in PRIVATE.BEL s.t. P resolves Q, answer(P)

S: ”The price is £123”

QUD-Based ISU Modeling

- Dealing with multiple issues:
  - if user asks Q, push Q on QUD and load plan for dealing with Q
  - if users asks Q' while system is dealing with Q, **throw out plan for Q** but Q remains on QUD
  - when Q' resolved, Q topmost on QUD will trigger reloading plan for dealing with Q
    - general rule: if SHARED.COM contains info resolving Q, don't ask Q
    - so any resolved questions in plan will be thrown out

QUD-Based ISU Modeling

U: I want price information [raise ?x.price(x)]

S: Where do you want to go?

U: London

S: When do you want to travel?

QUD=<?x.dept-month(x), ?x.price(x)>

U: Do I need a Visa? [raise ?visa]

QUD=<?visa, ?x.dept-date(x), ?x.price(x)>

S: Where are you travelling from?

U: Gothenburg

S: No, you don’t need a Visa.

ISSUES=<?x.dept-month(x), ?x.price(x)>

PLAN empty; QUD empty

QUD-Based ISU Modeling

S: No, you don't need a Visa.

QUD=<?x.dept-date(x), ?x.price(x)>

U: OK, I want to leave in April [answer dept-month(april)]

QUD=<?x.price(x)>

PLAN empty, so reload plan for dealing with ?x.price(x)

Throw out all question which have already been resolved; raise the first unresolved question on plan

S: What day do you want to leave?

…

QUD-Based ISU Modeling

- Advantages:
  - Generic approach to dialogue modeling
  - Handling various dialogue phenomena
    - Accommodation (“overanswering”)
    - Reraising of issues
    - Task switching, sharing information across tasks
    - Various dialogue genres (e.g., negotiation, tutoring…)
    - …
- Disadvantages:
  - Static dialogue plans (recipes)

    --> integrate with plan-based approaches, where focus is on task planning

Reusability: Toolkits

- CSLU toolkit was a pioneer FSM-based toolkit
- DialogOS: http://www.clt-st.de/produkte-losungen/dialogos/
- ISU-based toolkits:
  - TrindiKit (Gothenburg U.) http://www.ling.gu.se/trindi/trindikit/
  - Dipper (U. of Edinburgh) http://www.ltg.ed.ac.uk/dipper/
  - MIDIKI (MITRE Corp.) http://midiki.sourgeforge.net/
- Olympus/RavenClaw http://wiki.speech.cs.cmu.edu/olympus/index.php/Olympus
- OpenDial https://code.google.com/p/opendial/
- SceneMaker:
  - http://www.dfki.de/~gebhard/
  - https://www.informatik.uni-augsburg.de/en/chairs/hcm/projects/tools/scenemaker/

Dialogue Policy Design

(slides based on Pinkal&Wolska 2012)

Dialogue Policy Design

- The general task of designing an dialogue model can be subdivided into two separate sub-tasks:
  - Specification of a fixed framework for the dialogue structure through global design decisions:
    - Specification of a set of dialogue states
    - Specification of the range of available actions of the dialogue system
    - over-all structure of the dialogue (e.g., information collection phase,database look-up, answer generation).
  - Specification of a dialogue policy: a decision procedure which decides at a certain point of the global model which concrete action out of a set of alternatives should be taken. Examples:
    - Grounding: Explicit grounding act/ implicit grounding act/ no grounding
    - Selection of presentation mode and modality for alternative user options.

Dialogue Policy Design

- Global design decisions: the specification of
  - a set of dialogue states S, the State Space (e.g., a set of nodes in a FSA, or a set of structured information states)
  - a set A of possible system actions, the Actions Set (transitions, ISU operations)
  - a range of admissible possible actions A(s) for each state s
- A dialogue policy is a function from states to actions
- How to find an optimal dialogue policy?

Determining Dialogue Policies

- How do we find the optimal dialogue policy?
  - Set alternative parameters by hand; examples:
    - Confidence threshold for grounding, dependent on the importance of the decision
    - Maximum number of items for which graphical display is appropriate (dependent on actual user situation)
  - Run full implemented system or WoZ experiment with human users, evaluate, modify or refine.
- Comment:
  - This is at present the typical solution in real-world applications.
  - Adaptive dialogue behaviour requires to take a large set of feature combinations into account. Costs are high, results not always optimal.
  - Is machine learning an alternative?

Supervised Dialogue Policy Learning

- Collect data from WoZ experiments, which are set up in a novel way:
  - Several wizards. No specific strategy on the wizards' behavior, but just a general instruction ("Help the user reach its goal!")
  - Derive a n-gram based statistical model from the data that proposes the most probable system move as the appropriate system reaction.
- Difficulties:
  - Learnt dialogue behaviour is locally consistent, global optimisation is not supported (due to n-gram constraint).
  - System reproduces the average wizards' behavior; it cannot assess the value of the decisions, or include novel, unrealized decisions.
  - Data sparseness.

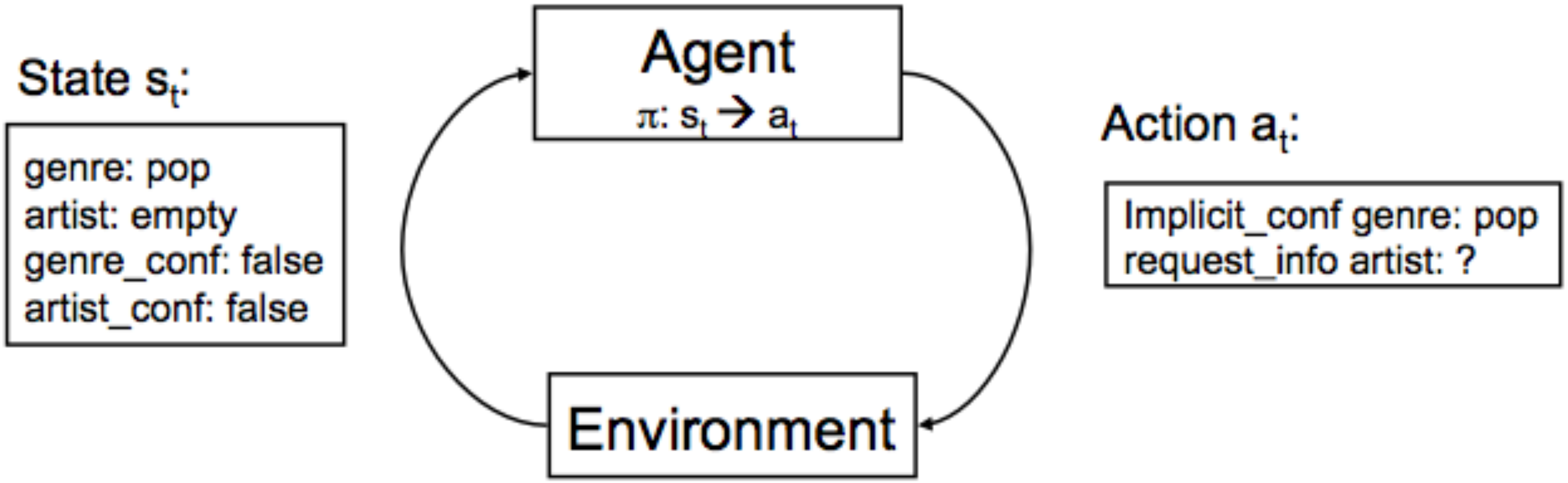
Reinforcement learning

- System learns optimal dialogue policy by executing dialogues and getting feedback on its performance: Reinforcement Learning.
- Reinforcement learning builds on the concept of Markov Decision Process, a framework for modeling decision-making by an agent in an environment whose behavior is (partly) random.
- The agent selects an action based on the current state (plus reward information).
- The environment emits information about the state it has adopted (and assigns a reward for the agent's last action).

Decision Process

- Agent interacts with a stochastic environment:
  - At time t, agent picks an action a_t (on the basis of its current state s_t and a strategy/policy π)
  - Execution of a_t (non-deterministically) influences the subsequent behaviour of the environment, which thus gets to state s_{t+1}.
  - According to strategy π, agent (deterministically) selects and executes the next action: $\pi(s_{t+1}) = a_{t+1}$

Decision Process: Example

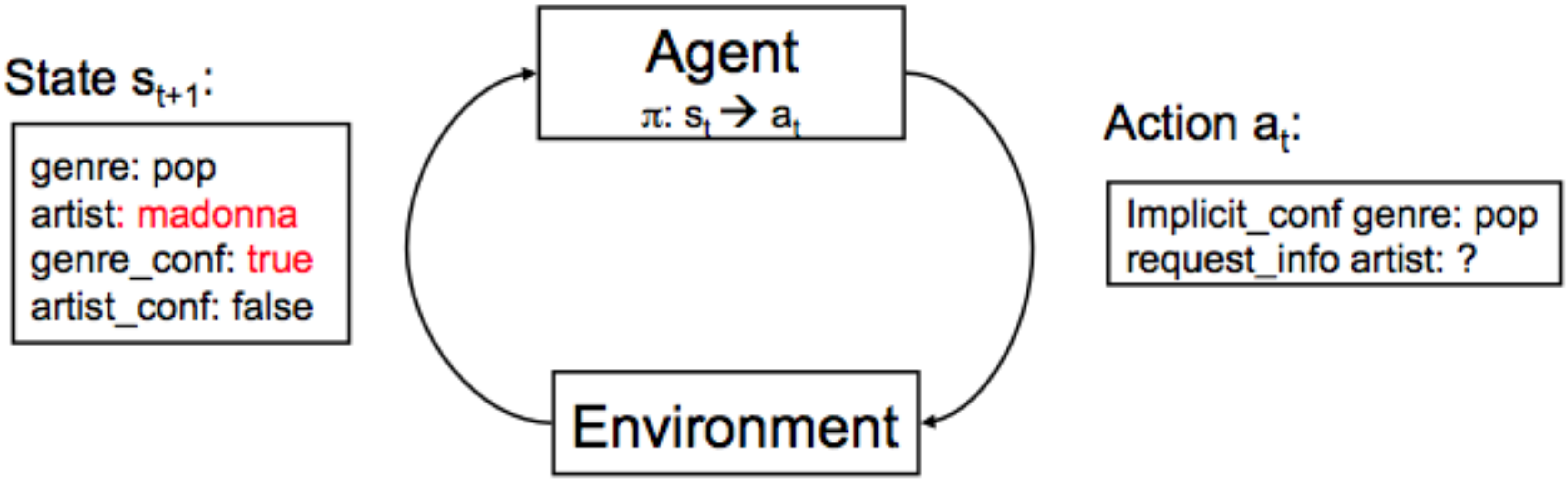


S1: What kind of music do you want to hear?

U1: Pop music.

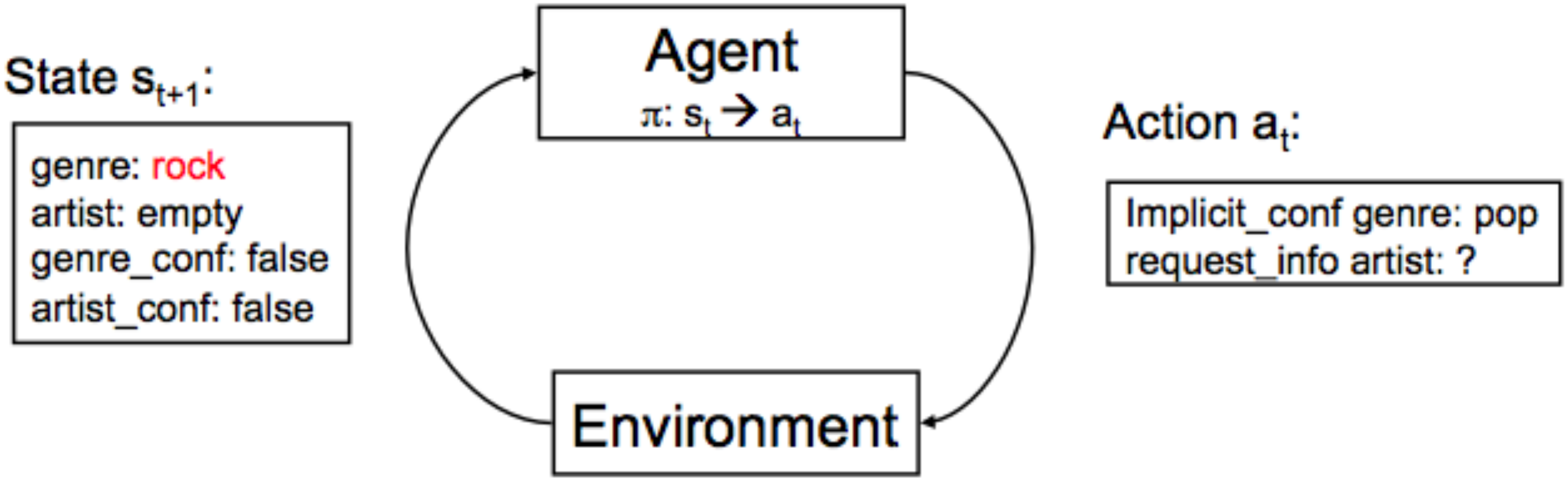
S2: By which artist do you want to hear pop music?

Decision Process: Example



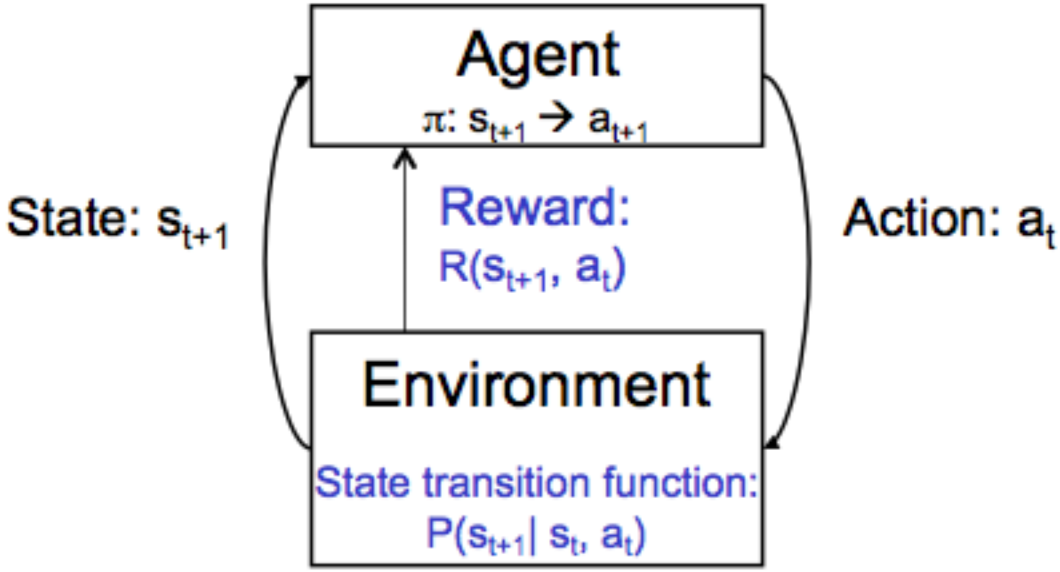
S1: What kind of music do you want to hear?
U1: Pop music.
S2: By which artist do you want to hear pop music?
U2: Madonna.

Decision Process: Example



S1: What kind of music do you want to hear?
U1: Pop music.
S2: By which artist do you want to hear pop music?
U2: Not pop, rock!

Modeling the Environment



Estimating P

- Produce a dialogue corpus using WoZ experiments. Then
- Learn n-gram probabilities of state transitions (user dialogue moves) from the corpus.

Rewards

- Immediate reward R(s,a) is locally assigned to state-action pairs.
  - To make an informed choice between two actions, local comparison of immediate rewards is insufficient.
  - The utility of an action strongly depends on its long-term effects: E.g., the utility of a dialogue contribution depends on the question whether it contributes to an over-all successful execution of the dialogue.
  - We have to assess the expected value of an action, by estimating its expected cumulative reward Q(s,a), which combines immediate effect and long-term effects of action a, executed in state s.

Designing R

- Determine immediate reward by hand, using intuition and experience.
  - Arbitrary at least to some degree.
- Assess reward for the dialogue as a whole via(SASSI style) user questionnaires. Only final state-action pairs get non-zero reward.
  - Sparse data problem: Human users assess quality of the full dialogue sequence they have gone through; no assessment of sequences which have not occurred in the data.
- Approximate user assessment through measurable features.
  - PARADISE scheme
- Model user: Simulation-based reinforcement learning

The PARADISE Questionnaire

- TTS Performance: Was the system easy to understand?
- ASR Performance: Did the system understand what you said?
- Task Ease: Was it easy to find the information you wanted?
- Interaction Pace: Was the pace of interaction with the system appropriate?
- User Expertise: Did you know what you could say at each point in the dialogue?
- System Response: How often was the system sluggish and slow to reply to you?
- Expected Behaviour: Did the system work the way you expected it to?
- Comparable Interface: How did the system's vioce interface compare to othersystems?
- Future Use: From your current experience with using the system, do you think you would use the system regularly?

How Realistic is RL

- A general problem for all dialogue learning methods is the large state space (exponentially growing with number of features/dimensions)
- So far, RL methods are not straightforwardly applicable for the design of dialogue systems with realistic complexity.
- Methods to get around the complexity trap:
  - State space reduction
  - „Sub-Strategy Learning“: Handcode most of the policy, and leave only special difficult design decisions for automatic optimisation

Current Topics/Challenges

- Adaptivity:
  - Systems need to be dynamically adaptive in a number of different ways: to the environments in which they are used (modality), to their user's preferences and needs (personalisation), and to changes in task and context.
- Learnability:
  - Systems need to be able to learn from interactions with users in order to provide an optimally usable interface that matches the current environment and user.
- Standardisation:
  - There is a need for a common set of standards to support re-usability for developers and to support usability for the users of spoken dialogue systems.
- Pervasive systems
  - Systems need to handle distributed dialogues (shifts to different dialogue situations / managers), concurrent dialogues (issues of co-ordination, synchronisation, redundancy); interaction model needs to be predominantly event-based (external events, opportunistic)