Dialogsysteme und VoiceXML

Christoph Ringlstetter/Prof. Klaus U. Schulz

Seminar im WS 2004/05

Sitzung III (02.11.2004): Christoph Ringlstetter Dialogmanagement

- Überblick zu den 3 Managementparadigmen
  - finite-state Systeme
  - frame-basierte Systeme
  - agentenbasierte Systeme
- Formularinterpretationsalgorithmus für framebasierte Systeme
- 2 Beispiele zum agentenbasierten Dialogmanagement

Animation: Sparda

DAMSL-Notation der Dialogakte im Spardadialog: Martina Pajic

1 Opening	Begrüßung :“Willkommen bei der Spardabank...“
2 Action- directive	Aufforderung: „Bitte notieren Sie sich ihre provisorischen Kundendaten...“
3 Opening explicit performative	"Guten Tag" "...Geburtstag" Persönliche Begrüßung des Kunden;
4 Statement	Gibt den aktuellen Kontostand aus;
5 Open-Option	Verschiedene Optionen werden angeboten: „Start“, Information“ und „Stichwort“ ;
6 Accept-part	Kunde wählt “Stichwort”;
7 Committing-speaker-future-action (Offer)	Er soll nun wählen unter den 10 verschiedenen Unterfunktionen von “Stichwort”;
8 Accept-part	Kunde wählt “Überweisung”;
9 Repeat-rephrase	Frage „Möchten sie eine Überweisung tätigen?“
10 Accept	Kunde antwortet mit „Ja.“;

DAMSL-Notation der Dialogakte im Spardadialog: Martina Pajic

• 11 info-request	Aufforderung: „Bitte geben sie die Bankleitzahl
•	des Empfängers an...“
• 12 Answer	Kunde nennt diese;
• 13 Correct- misspeaking	„Diese BLZ existiert leider nicht!“
• 14 Infor-Request	Aufforderung: „Bitte geben sie die BLZ erneut
•	an...“
• 15 Answer	Kunde nennt diese erneut;
• 16 info-request	Aufforderung: „Bitte geben Sie nun die
•	Kontonummer des Empfängers an...“
• 17 Answer	Kunde nennt diese;
• 18 Info-Request	Aufforderung: „Bitte buchstabieren sie nun den
•	Namen des Empfängers...“
• 19 Answer	Kunde buchstabiert diesen;
• 20 Signal-non-undersdtanding	System versteht die Eingabe nicht;

DAMSL-Notation der Dialogakte im Spardadialog: Martina Pajic

• 21 Info_request	Aufforderung: „Bitte buchstabieren sie erneut den Namen des Empfängers...“
•	
• 22 Answer	Kunde buchstabiert diesen noch einmal;
• 23 Open-Option	System bietet Option „Verwendungszweck“ an;
• 24 Reject	Kunde möchte keinen Verwendungszweck angeben;
• 25 Repeat-rephrase	Frage „Sind folgende Daten korrekt: BLZ...,KTN...,Name...?“
•	
• 26 Accept	Kunde antwortet mit „Ja.“;
• 27 Signal-understanding	und Bestätigt die Überweisung auf das Empfänger-
• Information-relations	konto;
• 28 Open-Option	Verschiedene Optionen werden wieder angeboten:
• „	Start“, Information“ und „Stichwort“ ;
• 29 Closing	Kunde sagt jedoch „Ende“;
• 30 Repeat-rephrase	Frage „Wollen sie das Menü jetzt verlassen...?“
• 31 Accept	Kunde antwortet mit „Ja.“;
• 32 Closing	System bedankt und verabschiedet sich.
•	

Fragen die die Sitzung beantworten soll:

- Was ist Dialogmanagment
- Wie wird Dialogmanagment gewährleistet/implementiert
- Gibt es einen Zusammenhang zwischen Komplexität des Managementmodells und der Taskkomplexität?
- Gibt es ein Modell das zu bevorzugen ist?

Definition: Dialogmanagement

- Hauptaufgabe des Dialogmanagements im Rahmen "praktischer Dialoge" ist es, die Aufgabe (task) des Dialogs durchzuführen
- Beispiel Datenbankabfrage:
  - Einholen aller notwendigen Informationen vom Benutzer, um eine externe DB-Anfrage durchzuführen
  - Durchführen der Kommunikation mit der Datenbank
  - Kommunizieren von Information zurück zum Benutzer
- Benutzerorientierte Aufgaben des Dialogmanagements:
  - fehlende Information erfragen
  - Benutzeranfragen zielführend verarbeiten
- Systemorientierte Aufgaben des Dialogmanagements:
  - Applikationen kontrollieren
  - Grounding: Missverständnisse erkennen und auflösen
  - Clarification: potentielle Fehler beheben
  - Teilaufgaben erkennen und abschließen: Aufgabenstrukturierung gewährleisten
  - bei aufgehobener Pipelinestruktur: Benutzereingaben vorhersagen und so die Spracherkennung verbessern

Dialogmanagement (i. w. S.)

- Umsetzung linguistischer Theorie aus:
  - Sprechaktforschung
  - Diskursforschung
  - Forschung zu Mensch-Mensch Dialogen
  - Forschungen zum Benutzermodell
- Kontrolltechnologie (Dialogmanagement i. e. S.):
  - Endliche Automaten
  - Framebasierte Slot-Filling-Systeme
  - Agentenbasierte Systeme
- Initiative-Steuerung:
  - systemdriven: während des gesamten Dialogs behält das System die Initiative
  - userdriven: der Benutzer behält ständig die Initiative
  - mixedinitiative: die Initiative wechselt
- Clarification/Grounding: Übereinstimmung zwischen System und Benutzer erzielen
- Externe Wissensaquisition: etwa Datenbankabfrage

Einsatz verschiedener Managementparadigmen nach der Komplexität von Dialogsystemen

- Gerätebedienung
  - Bedienung verschiedener Funktionen im Auto mit gesprochener Sprache: finite-state
- Informationsbeschaffung
  - einfache Datenbankabfrage: finite-state
- Beschaffung strukturierter Dienstleistungen
  - Pizza, Fahrkarten: finite-state und oft frame-basiert
- kombinierte Aufgaben
  - Information und Bestellung in einem System: frame-basiert
  - Buchung zweier voneinander abhängiger Dinge wie Mietwagen/Motel: frame-basiert, "agenten-basiert"
- komplexe Aufgaben
  - Fix-circuit-shop-System: Reparatur von elektr. Schaltkreisen: agenten-basiert: Theorembeweiser
  - Planungssystem mit dynamischen Anteilen (Trips System zur Planung des Einsatzes medizinischer Notfallresourcen wje etwa Rettungswägen): agenten-basiet: conversationeller Agent

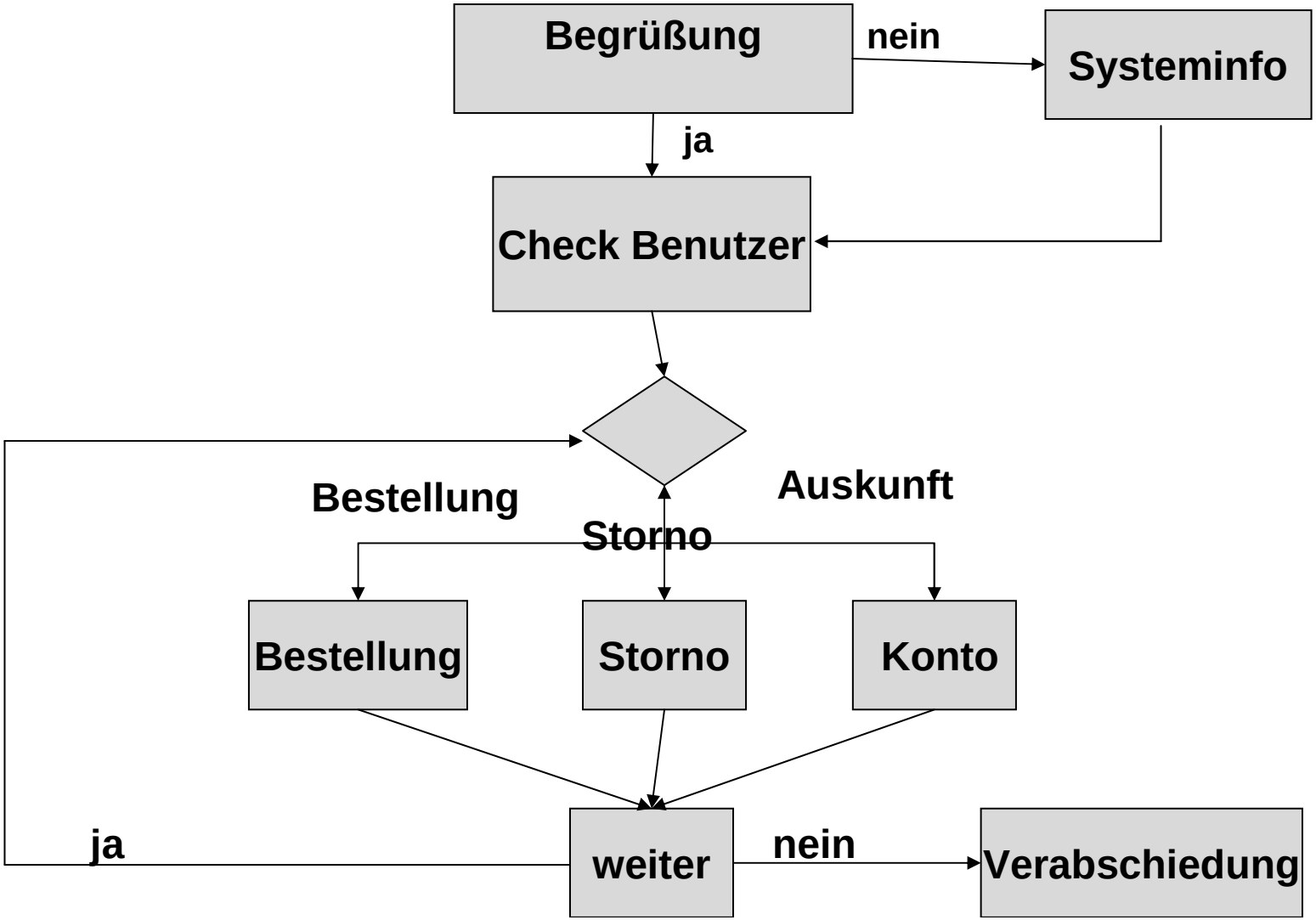
Dialogkontrolle mit deterministischen endlichen Automaten

- Die Dialogstruktur kann als Zustandsübergangsgraph repräsentiert werden
  - Knoten: Systemfragen, Systemausgaben, Systemaktionen
  - Kanten: alle möglichen Pfade im Netzwerk gelabelt mit den Benutzeräußerungen
  - $\Rightarrow$ Modell aller legalen Dialoge kann in einem Zustandsgraphen abgebildet werden
  - $\Rightarrow$ Da alle Wege in einem deterministischen endlichen Automaten zum Zeitpunkt der Systementwicklung festliegen und jeder Übergang mit einem festgelegten Input verbunden ist kann von einer expliziten Dialogkontrolle gar nicht gesprochen werden. Der Dialog ist starr, alle Reaktionen für jeden user-input stehen fest

"automatic bookservice": Systemprompts und Benutzereingaben für das Primärszenario

- S: Herzlich willkommen beim Voicezugang des bookservice. Wissen Sie wie sie das Systembenutzen sollen?
- U: ja/nein -> Verzweigung zu Systeminfo
- S: Bitte nennen sie Ihre geheime Benutzerkennzahl
- U: Nennt richtige Zahl
- S: Falls sie Bestellen wollen sagen sie *Bestellung*, falls sie eine Bestellung stornieren wollen sagen Sie *Storno*, falls sie Auskunft über ihre Bestellungen benötigen sagen sie *Auskunft*
- *U:* Nennt eine Funktionalität
- S/U: Funktionalität wird ausgeführt
- S: Wollen Sie weitere Funktionen durchführen?
- U*:* ja/nein -> Rückverzweigung oder Verzweigung zur Verabschiedung
- *S:* Vielen Dank für ihren Besuch bei/ ihre Bestellung bei etc.

Beispiel für FSA-Dialogkontrolle: Voice Schnittstelle "automatic bookservice"



"automatic bookservice": Charakteristik

- Offensichtlich ist das Primärszenario zu jedem Zeitpunkt in einem wohldefinierten Zustand.
- Alle Übergänge sind deterministisch
- => sehr gut als FSA darstellbar
- Sekundärszenarien wie: Clarification, Hilfefunktion usw. können bei jedem Zustand eingefügt werden, ohne dass sich die Gesamtcharakteristik verändert

Vorteile der Dialolgkontrolle mit deterministischen Automaten

- Vorteile:
  - natürliche Art systemgeleitete Dialoge zu repräsentieren
  - gut geeignet vollständig vorhersehbaren Informationsaustauschs zu modellieren,

    das gesamte Dialogmodell ist zur Entwicklungszeit definiert
  - oft ist es möglich den Dialogfluß graphisch zu repräsentieren: **natürliches Entwicklerinterface**
  - Nutzung FSA für: simple Aufgaben mit flacher Menü-Struktur und kurzen Optionenlisten

- Aufgrund der Einfachheit v. a. der Sprachverstehenskomponente und der möglichen Einschränkung der Zahl an aktiven HMMs für die Erkennung im Moment in den meisten kommerziellen Systemen genutzt (z. B. auch im Sparda Dialog)

Nachteile der Dialolgkontrolle mit deterministischen Automaten

- ungeeignet für schlecht strukturierte Aufgaben gibt es keine natürliche Ordnung der Zustände => jeder Zustand kann jedem folgen => kombinatorische Explosion: die Automaten werden zu groß
- ungeeignet für verschiedene Abstraktionslevels der ausgetauschten Informationen
- ungeeignet für komplexe Abhängigkeiten zwischen den Informationseinheiten: Bsp.: Flugreservation: eigentlich gut strukturiert, aber Parameterabhängigkeiten etwa zwischen Sonderangbeboten und Flugverfügbarkeit. Oft sind tiefe Backtracking-Funktionalitäten notwendig, die über FSA-Systeme nicht realisiert werden können.
- Systeme sind unflexibel da vollständig system-geleitet, keine Pfadabweichungen möglich.
- Kontext von Äußerungen kann nicht genutzt werden
- Grounding und Repair sind sehr starr, müssen nach jedem Turn durchgeführt werden. Spätere Korrekturen durch den User sind kaum möglich
- Daten zur Zustandsexplosion: Nach Aust/Oerder(1995) zum Philips Zugbuchungssystem bei Finite State Implementierung "10000de von Übergängen" notwendig
- Aufgaben, die Verhandlungen (Negotiations) erfordern können nicht mit deterministischen Automaten implementiert werden, da das Ergebnis vorher ja nicht feststeht. Beispiel ist das Trains-Projekt: Ein Plan wird während des Dialogs unter dynamischen constraints entwickelt

Charakterisierung der FSA-Kontrolle

- empirische Studie zu einem alternativ implementierten Telefonauskunftssystem (vgl. Mc Tear 2002): längere Transaktionszeiten, mehr turns für das FSA-System als für das mixed initiative System mit einer komplexeren NLP Komponente
- aber: durch ein strikteres Modell für die Spracherkennung gibt es für FSA Modelle weniger Erkennungsfehler => weniger Clarification-Turns
- Ergebnis: systemgeführtes FSA-System war nicht langsamer und lag nicht schlechter in der subjektiven Benutzerzufriedenheit. Weitere ähnliche Studien untermauern das Ergebnis.

Framebasierte Dialogkontrolle

- Unterschied zu Automatentechniken: Die Äußerungen welche die Felder eines Formulars füllen, müssen nicht in einer vorbestimmten Reihenfolge auftreten
- Notwendige Komponenten:
  - Frame: führt Buch über Informationsfragmente die bereits geäußert worden sind
  - Erweiterte Erkennungsgrammatik für flexiblere Benutzeräußerungen
  - Dialogkontrollalgorithmus, der die nächsten Systemaktionen auf Grundlage der vorliegenden Frameinhalte festlegt
- Vorteil: flexibler, mixed initiative eingeschränkt möglich
- Technologie wird z. B. von VoiceXML umgesetzt

Dialogmanagment mit Frames

- Ein Frame oder Template (<form> - Element in VXML) kodiert die Items oder Slots (<field> - Element in VXML) die das System mit Informationen füllen muss.
- Die Systemfragen hängen an den Slots
- Beispiel für das SpeechMania Zugauskunftsystem von Philips (1994):
  - Fragen werden zusammen mit Vorbedingungen kodiert:
    - Frage nach Abfahrtsbahnhof & condition: unknown(origin)
    - Frage nach Zielbahnhof & condition: unknown(destination)
  - Algorithmus für das Dialogmanagment: Die nächste Frage wird aus der Menge der Fragen gestellt, deren Vorbedingungen wahr sind. Bei Fragenkonkurrenz folgende keywords zur Prioritätssteuerung:
    - (1) ONCE: initiale Begrüßung
    - (2) MULTIPLE: es wurde mehr als ein Wert von der Erkennung zurückgegeben
    - (3) VERIFIABLE: Wert ist noch nicht vom Benutzer bestätigt
    - (4) UNDEFINED: Variable noch unbelegt
  - Konsequenz: Ambiguitäten werden vor Verifikationen und diese vor Abfragen neuer Informationen durchgeführt

    Der Dialog wird gesteuert, ohne zur Entwicklungszeit einen Pfad durch den Dialog festzulegen

Dialogmanagment in VoiceXML

- Ein Formular in VoiceXML besteht aus Feldern + Kontrolleinheiten
  - Feld: sammelt Informationen vom Benutzer via Spracheingabe oder DMTF(Dual Tone Multifrequency)
  - Kontrolleinheiten: Sequenzen prozeduraler Statements
- Die Steuerung des Dialogs erfolgt über den nachfolgenden Formularinterpretationsalgorithmus. Dieser legt entsprechend ihrer guard-Bedingung fest, welche Felder besucht werden sollen

Der Formularinterpretationsalgorithmus (FIA) in VoiceXML - informell

- Hauptschleife mit 3 Phasen:
  - **select**: Top-Down im aktiven VoiceXML Dokument wird das erste nicht ausgefüllte Formular ausgewählt, dessen guard-Bedingung offen ist (Ausnahme: es gab eine goto-Sprunganweisung)
  - **collect:** Das ausgewählte Formular wird besucht. Der nachfolgende Promtausgabealgorithmus wird auf die Prompts (=Systemäußerungen) des Formulars angewendet, die Eingabegrammatik des Formulars wird aktiviert. Der Algorithmus wartet auf Eingaben des Benutzers
  - **process:** Eingabeauswertung indem die Felder des Formulars gemäß der aktiven Grammatik gefüllt werden. Gefüllte Elemente werden beispielsweise zur Eingabevalidierung aufgerufen. Die process-Phase endet, wenn keine Items mehr augewählt werden können oder ein <goto> erreicht wird.

Auswahl der <prompt>-Elemente im FIA während der collect Phase

- Als "prompt-Queue" wird eine Liste aller prompts in der Reihenfolge ihres Auftretens im Formular angelegt
- Alle prompts deren guard Bedingung geschlossen ist werden aus der Queue gestrichen
- Vergleich der count Werte der einzelnen prompts (wie oft kann ein prompt ausgeführt werden) mit dem Formular count (wie oft wurde das Formular ausgeführt)
- Abspielen der übrig gebliebenen prompts Top-Down nach dem Auftreten im Formular

Weitere Varianten framebasierter Systeme

- E-Forms: System für das Angebot von Gebrauchtfahrzeugen
  - Slots können benutzerabhängig (adaptiv) unterschiedliche Prioritäten haben: unterschiedliches Gewicht für Farbe, Modell
  - Systemverhalten wird u. a. durch die Zahl der zurückgelieferten Fahrzeuge gesteuert: je nach Zahl wird der Benutzer aufgefordert spezifischer oder allgemeiner zu werden

- Dialogkontrolltabelle im MIT Mercury-Flugreservationssystem (inzwischen > 250 Regeln) durch Management von state-variablen
  - !source → prompt_source
  - !destination → prompt_destination
  - !airline → prompt_airline
  - !date → prompt_date
  - nprompts >1 → mark_multiple
  - nprompts = 0 → retrieve flights

- Schemas: Dialogkontrolle im Carnegie Mellon Communicator System:
  - Führe falls vorhanden globale Systemaktionen aus
  - Falls Clarification aktiv ist führe sie durch
  - Falls Daten präsentiert werden müssen tue dies
  - Falls externe Daten aquiriert werden müssen tue dies
  - Falls der aktive Frame noch nicht komplett ist äußere die entsprechenden Feldprompts
  - Prüfe andere aktive Frames auf Vollständigkeit
  - Erzeuge den Schlussframe
- Im System ist topic-shifting implementiert, der Dialog wird als offener Baum implementiert. Formulare können dynamisch verändert werden.

Vorteile der Dialolgkontrolle mit Frames

- Höhere Flexibilität für den Benutzer
- Multiples Slot-Füllen
- Kürzere Dialogzeiten
- Mixed-Initiative Steuerung möglich
- Durch VoiceXML allgemeine Spezifikation vorhanden

Nachteile der Dialolgkontrolle mit Frames

- Begrenzter Kontext der über die nächste Aktion des Systems entscheidet:
  - letzte Benutzereingabe
  - Status der Slots
  - einfache Rangfolge von Prioritäten
- Kenntnisstand des Benutzers kann nicht oder nur sehr rudimentär modelliert werden
- Verhandeln oder kollaboratives Planen können nicht modelliert werden
- Gesamtüberblick über das Systemverhalten ist aufgrund z. T. komplexer Regeln nicht möglich: "Welche Regel feuert wann?"

Agentenbasierte Dialogkontrolle

- Dialog als Kollaboration zwischen intelligenten Agenten um ein Problem oder eine Aufgabe zu lösen
  - kooperative Antworttechniken
  - Schlußfolgerungen über Aktionen und Vorstellungen von sich selbst und anderen
  - fortgeschrittene Fehlererkennung und Fehlerkorrektur
  - Erwartungen bilden, um die nächsten Äußerungen des Benutzers zu interpretieren
- Verwendung von Techniken aus der KI
  - Unterbrechbare Theorembeweiser
  - Planning-Techniken
  - verteilte Architekturen
  - konversationelle Agenten
- Charakteristik:
  - volle mixed initiative
  - theoretisch sind komplexe Systeme implementierbar, die dynamisch/kollaborativ komplexe Aufgaben lösen
  - deutlicher an der linguistischen Theorie wie Sprechakttheorie, Conversational Games Theorie, Kohärenztheorie etc. orientiert
  - bislang lediglich akademische Systeme realisiert

Exkurs: Definition des Agentenbegriffs

- Eine allgemeine Definition gibt es nicht:
- "Ein Agent ist ein System das situiert ist in einer Umgebung, das diese Umgebung wahrnimmt und ihr bezüglich handelt womit es beeinflußt was es in Zukunft wahrnimmt" (Franklin: 1997)
- Charakteristische Eigenschaften sollen sein: situiert, reaktiv, autonom, sozial, rational, antropomorph
- z. B.: Allen: "deliberativ-reaktive" Agentenstruktur = Agent der berät und sein Verhalten nicht nur nach einem Planungsmechanismus richtet sondern auch nach dem was während eines Dialogs "passiert".

Nachtrag: Dialogmanagement und Kohärenz

- Fundamentale Idee: Mit einem Dialog ist eine Zweck verbunden (DP: dialogue purpose), den der Agent hegt, der den Dialog angefangen hat. Weiterhin hat jedes Dialogsegment innerhalb eines Dialogs einen Zweck (DSP: dialogue segment purpose), eine Rolle hinsichtlich des Gesamtdialogs.
- Beispiele für Intentionen:
  - einen Agenten dazu bringen eine physische Aufgabe auszuführen
  - einen Agenten etwas Glauben machen
  - einem Agenten ein Objekt zeigen (physisch, imaginär, Plan, Event)
  - einem Agenten eine Objekteigenschaft zeigen
- Zur hierarchischen Ordnung von Intentionen leiten Groz/Sidner Kohärenzbeziehungen ab wie z. B.: Dominanz und Präzedenz.
- Aus diesen Kohärenzbeziehungen lässt sich die Dialogstruktur ableiten.

Dialogkontrolle mit unterbrechbaren Theorembeweisern

- Lösungen werden auf der Basis von Aufgabenschritten, die die aktuelle Situation und den aktuellen Wissensstand beachten entwickelt. Goal-Abarbeitung etwa im Rahmen eines Prolog-Beweissystems
- Per Subdialog wird versucht fehlende Axiome eines Zieles zu ermitteln: "Missing Axiom Theory". Diese werden in den partiell erfüllten Beweis eingefügt
- Als Maschinerie wird ein unterbrechbarer Theorembeweiser (interuptable Prolog Simulator) eingesetzt
- Ziel wird zu Beginn des Dialogs explizit gesetzt
- Beispiel: Circuit-Fix-It-Shop-System

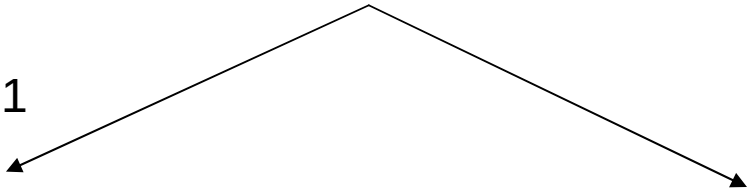
Beispielsequenz circuit-fix-it-shop-system I

- Der Dialogmanager bekommt eine Aufgabe vom Domainprozessor und wählt aus, welches Ziel (goal) bewiesen werden muss
- Der Dialogmanager entscheidet, wann der Theorembeweiser aktiv wird, ob Sprache benutzt werden kann um fehlende Axiome zu aquirieren und wann Axiome dynamisch verändert werden müssen
- Beispiel: Das System muss herausfinden ob es ein Kabel zwischen den Kontakten 84 und 99 gibt:

Beispielsequenz circuit-fix-it-shop-system II

- goal(computer, learn(ext_know(phys_state(prop(wire(84,99),exist,X),true))))
- GADL(Goal and Action Description Language)
- axiom(phys_state(prop(wire(84,99),exist,absent),true)) → wire is absent
-
-
- do(action(add,wire(84,99)))

- inserted subgoal 1



- learn to do add
- ("I need help")

missing axiom
("add a wire")

Planbasierte Ansätze

- Dialogäußerungen werden behandelt wie Aktionen in einem Plansystem
- Schlüsselelement: Modellieren der Äußerungen als Sprechakte (verarbeitung in einem Modell mit Rollen, Preconditions,Constraints und Effekten gemäß dem BDI-Paradigma : siehe Sitzung 2)
- Schwierigkeiten:
  - Plan ist zu Beginn nicht explizit vorhanden
  - um den Plan eines Benutzers zu ergründen muss der zuhörende Agent den "kommunikativen Akt" der sprachlichen Äußerung erkennen und in ein Planschema einfügen
  - Für komplexe Anwendungen entsteht ein kombinatorisches Problem: Daher sind Planning-Algorithmen am besten für restriktive Domains geeignet: "in which the reasoning is kept to managable proportions "(Mc Tear) → Widerspruch zum eigentlichen Ziel socher Ansätze

Material der Planung: Parseroutput ist eine Sequenz von Sprechakten

- Bsp.: (Allen et al.)
- Sprecher: "Okay let's take the train from Detroit to Washington"
- ASR → System: "Okay let's send contain from Detroit to Washington"
- Sprechaktsequenz aus bottom up parser und einer syntaktisch/semantischen Grammatik:
  - CONFIRMATION ("okay")
  - TELL (interpretierbare Wortsequenz "let's send contain")
  - TELL (Erwähnung einer Route: "from Detroit to Washington")
- Parsingergebnis reicht dem System bei entsprechendem Dialogkontext aus um mit der Sequenz ohne Klärungssubdialog weiterzuarbeiten

Beispiel: Trains/Trips System zur Verwaltung von Notfallressourcen

- User: Wir müssen die Frau von München-Flughafen nach München bekommen.
- System: ok.
- User: Welche Fahrzeuge sind verfügbar?
- System: Es gibt Ambulanzen in Moosburg und Freising.
- User: Ok. Benutze eine aus Freising.
- System: Wissens Sie, dass auf der A9 ab Eching wegen des Stadionneubaus ein Stau ist?
- User: Oh. Benutzen wir stattdessen die Bundesstraße!
- System: Ok. Ich informiere die Crew.

(nach: Allen et a. 2001a, S. 30)

Trains/Trips: Interpretational Agent

- Das System muss fähig sein, zu schließen, dass der Stau auf der A9 ein mögliches Hindernis für den intendierten Plan des Benutzers darstellt.
- Trips Diskurskontext (Interpretation Manager):
  - Modell der erwähnten Entitäten des Dialogs auf die im Verlauf referiert werden kann
  - Strukturrepräsentation der unmittelbar vorangehenden Äußerung
  - Status des Turns: Wer hat das Rederecht"
  - Diskurshistorie der Dialogakte mit Vermerk ob sie einem Grounding unterzogen worden sind
  - Verzeichnis aktueller Verpflichtungen (Obligations)

Trains/Trips: Behavioural Agent

- Interpretiert Benutzeräußerungen und Aktionen hinsichtlich problemlösender Akte
- Führt Buch über Systemziele und Obligationen
- Erfaßt Veränderungen des Zustandes der Welt: dynamische Veränderungen während der Dialogzeit können modelliert werden: z. B. Stau

Trains/Trips: Obligationenmodell

- Das Modell der Obligationen wurde eingeführt um zeigen zu können, warum das System eine Frage des Benutzers beantwortet:
- Ursprüngliche Lösung: es mussten shared plans und damit übertrieben kolaborative Einstellung beim Agenten angenommen werden: B fragt etwas also ist es B's Ziel eine Antwort zu finden -> Ziel bei A die Antwort auf B's Frage zu finden
- Lösung: Aufweichung des strengen Planparadigmas und dessen Erweiterung um "Obligationen": definiert über einen Modaloperator "permissible"
- Es besteht eine Vielzahl von Möglichkeiten Obligationen in den deliberativen Prozess einzubauen, je nachdem wieviel Gewicht sie besitzen sollen

Obligationenmodell: Übersicht

Quelle der Obligation	geforderte Aktion
S1 akzeptiert oder verspricht A	S1 realisiert A
S1 fordert A von S2	S2 akzeptiert A oder weist A zurück
S1 stellt J/N-Frage ob P	S2 antwortet if(P)
S1 stellt W-Frage P(x)	S2 informiert über P
Äußerung nicht verstanden oder nicht korrekt	Reparatur der Äußerung

Trains/Trips: Ablauf-Beispiel I

-> ab Äußerung: "Welche Fahrzeuge sind verfügbar?"

1. Spracherkenner merkt dass der Benutzer gesprochen hat
2. Interpretationsmanager (IM) interpretiert das als Turn-Taking-Event: Der Turn ist beim Benutzer
3. Sobald der Turn vollständig ist (Pause tritt ein) interpretiert das der IM als Turn-Releasing-Event
4. Der Generation Manager (GM) plant eine Antwort
5. IM erhält die logische Form der Benutzeräußerung und interpretiert sie im Kontext in Kolaboration mit dem Task-Manager: Der Term "Fahrzeuge" wird zu "Ambulanzen" im Sinne einer semantischen Grammatik, "verfügbar" sind Ambulanzen die nicht mit einer anderen Aufgabe beschäftigt sind.
6. Der IM macht ein Update vom Diskurskontext mit einer Repräsentation der Benutzeräußerung und einer Meldung, dass das System eine neue Verpflichtung hat eine Benutzerfrage zu beantworten

Trains/Trips: Ablauf-Beispiel II

7. Die anderen Systemkomponenten werden über die neue Obligation informiert
8. Der GM kann die Benutzerfrage nicht ohne Antwort vom Behavioural Agent(BA) beantworten. Geht in einen waiting state
9. IM sendet eine Message zu BA dass der Benutzer ein "Problem" initiiert hat
10. BA initialisiert das "Problem" und findet passende Ambulanzen in Freising und Moosburg: Optionen
    1. Frage beantworten + Antwort an GM senden
    2. Request Clarification: der BA kann nicht entscheiden welches die passendste Ambulanz ist -> Klärungssubdialog
    3. Failure: BA findet keine Ressource und meldet dies
    4. Ignore: BA hat etwas dringenderes im Plan und verschiebt die Obligation

Trains/Trips: Ablauf-Beispiel III

11. GM konstruiert eine Antwort
12. GM informiert den Diskurskontext dass versucht wurde die Obligation zu erfüllen
13. Update des Diskurskontextes
14. IM stellt Erwartungen über das Benutzerverhalten an: z. B. Obligation des Benutzers auf die Antwort zu reagieren
15. Wenn die Benutzerreaktion kommt: "Ok benutze eine aus Freising", weiss das System dass die Obligation erfolgreich erfüllt wurde
16. IM interpretiert die Äußerung und stellt fest, dass ein Teilziel des Plans, nämlich eine passende Ambulanz zu finden erfüllt ist und untersucht die Strecke...

Trains/Trips: Diskurs-Algorithmus

```
while Konversation noch nicht beendet
    if System hat Obligationen
    then kümmere dich um die Obligationen
    else if System hat den turn
    then   if System hat Konversationsziele
               then call Spracherzeugung(GM) und erstelle eine Äußerung
               else  if es gibt Konversationsmaterial ohne Grounding
               then Grounding der Situation
               else if ein Vorschlag ist nicht akzeptiert
               then erwäge Vorschlag
               else if high-Level Ziele sind unerreicht
               then address Ziele
               else überlasse den Turn dem Benutzer oder schlage das Ende der
                    Konversation
    else if niemand hat den Turn
    then nehme den Turn
    else if lange Pause
    then nehme den Turn
```

Interpretation des Diskursalgorithmus

- Der Algorithmus stellt ein "deliberativ-reaktives" Modell her, wo der Agent über seine Diskursverpflichtungen und seine Obligationen folgert
- Relaxed conversational Style: Der Agent kümmert sich zuerst um seine Obligationen und dann um seine "eigenen" Ziele
- Im System sind Fälle modellierbar, wo sich der Agent "weniger kooperativ" verhält und Obligationen durch Zurückweisung von Anfragen etc. "erfüllt".

Vorteile der "agentenbasierten" Dialolgkontrolle

- Kollaborative Systeme (konversationelle Systeme) erfordern eine komplexe Managementtechnologie. Finite-state oder frame-basierte Techniken sind nicht ausreichend
- Verhandlung über eine Aufgabe: System und Benutzer stellen Fragen, nehmen Korrekturen vor, machen Vorschläge ...
- In solchen echt mixed-initiative Systemen (level 3: Eröffnung von subtasks durch User und System) muss das Dialogsystem explizit über die Aufgabe (task) und den erreichten Durchführungsgrad folgern, über den Dialogstand sowie ev. über eigene Vorstellungen und solche des Benutzers

Nachteile der "agentenbasierten" Dialolgkontrolle

- Ressourcenbedarf deutlich höher: sowohl quantitativ(Rechnerleistung, Programmieraufwand) als auch qualitativ (Komplexität der zu lösenden Probleme) v.a. Modellierung von Intentionen + höherer Aufwand bzgl. der NLP Komponenten
- Kaum Reusability, kaum generische Tools wie Galaxy-Architektur oder VoiceXML
- Trotz des behaupteten Gegenteils: oft totale task-Abhängigkeit. Vgl. zur task-Abhängigkeit von KI-Systemen schon Weizenbaum (1976) zum "General Problem Solver von Newell/Simon: Weizenbaum: "Die Macht der Computer und die Ohnmacht der Vernunft"

Zusammenfassung: Dialogmanagment

Features/ Dialogstrategie	Finite-State-basiert	Frame-basiert	Agenten-basiert
Input	einzelne Wörter oder Phrasen	Natürliche Sprache mit Konzept-Spotting	Beschränkter Sprachausschnitt
Verifikation	Explizite Bestätigung nach jedem Input oder am Ende	Explizite und implizite Bestätigung	Grounding
Dialogmodell	Informationszustand implizit in den Dialogzuständen repräsentiert Dialogkontrolle explizit durch das Zustandsdiagramm repräsenti	Ínformationszustand explizit durch Frames repräsentiert Dialogkontrolle durch den Kontrollalgorithmus repräsentiert	BDI – Modell des Systems. Dialoghistorie und Dialogkontext
Benutzermodell	Einfaches Modell von Benutzercharakteristiken, festes Modell von Benutzeräußerungen	Einfaches Modell von User-Präferenzen	BDI – Modell vom Benutzer

Aufgabe 3

- Gehen sie auf die Webseite des VoiceXML Interpreters Optimtak www.optimsys.cz
- Laden sie das Softwarepaket Optimtalk herunter
- Schreiben Sie ein "Hello World"-Skript (vgl. S. 3 VoiceXML-Skript)
- Achtung Header muss für die neue Version so sein:
- <?xml version="1.0" encoding="ISO-8859-1"?>

  <vxml version="2.0" xmlns="http://www.w3.org/2001/vxml">
- Führen Sie das Programm mit dem Interpreter aus. Aufruf: Optimtalk/bin/optimtalk_test HelloWorld.xml

Vielen Dank für Ihre Aufmerksamkeit