CAPITOLUL VIII

METODE FORMALE ÎN IAD. REPREZENTAREA BAZATĂ PE LOGICĂ ŞI RAŢIONAMENTUL ÎN SISTEMELE BAZATE PE AGENŢI

Tehnologia agenţilor inteligenţi şi a sistemelor multi-agent pare să modifice radical, în ultima vreme, modul în care sistemele complexe şi distribuite sunt conceptualizate şi implementate. Problema reprezentării bazate pe logică şi a raţionamentului în sistemele bazate pe agenţi este un aspect important al domeniului, în care metodele formale joacă un rol hotărâtor.

În legătură cu formalizarea, două aspecte sunt importante:

- cum pot fi utilizate metodele formale pentru descrierea şi raţionamentul despre agenţi şi comportamentul lor;
- cum oferă metodele formale un mecanism de a demonstra că sistemele bazate pe agenţi (în maniera în care au fost implementate) sunt corecte în raport cu specificaţiile acestora.

În această secţiune, aspectele anterioare sunt discutate alegând ca şi studiu de caz arhitecturile OCI (secţiunea 3.2.3), a căror formalizare este extrem de puternică.

8.1 Ingineria soft a sistemelor bazate pe agenţi

Una din probemele controversate în literatura de specialitate este în ce măsură domeniul Agenţilor Inteligenţi ţine de Inteligenţa Artificială şi în ce măsură ţine de Ingineria Soft. O. Etzioni spunea că: "Agenţii Inteligenţi sunt 99% inginerie soft şi doar 1% inteligenţă artificială".

Agenţii Inteligenţi sunt priviţi, în general, ca sisteme de decizie autonome, care percep şi acţionează într-un anumit mediu. Problema construirii sistemelor bazate pe agenţi poate fi considerată ca o construcţie a ingineriei soft. Astfel încât, problemele cheie ale ingineriei soft, precum specificarea, rafinarea/implementarea, verificarea sistemelor bazate pe agenţi vor trebui luate în considerare. O provocare majoră a Inteligenţei Artificiale, şi a informaticii, în general, este de a asigura dezvoltarea corectă (atât ca specificaţii, cât şi ca implementare) a sistemelor bazate pe agenţi.

Este cunoscut faptul că metodele formale oferă o înţelegere a sistemelor proiectate la un nivel mai înalt decât implementarea lor specifică. Ele pot furniza o modalitate deosebit de importantă de a asista specificaţiile de depanare şi de a valida implementările sistemelor în raport cu specificaţiile lor precise.

8.1.1 Tehnologia bazată pe agenţi

Un *sistem bazat pe agenţi* este un sistem în care abstractizarea principală este aceea de *agent*. După cum am arătat şi în Capitolul I, un *agent* este un sistem care întruneşte următoarele proprietăţi:

- *autonomie*: agenţii îşi încapsulează starea şi iau decizii despre ceea ce fac pe baza acestei stări, fără intervenţia directă a altora;

- *reactivitate*: agenţii sunt situaţi într-un anumit mediu (o lume fizică, utilizator prin intermediul unei interfeţe grafice, o colecţie de agenţi, Internet-ul), sunt capabili să *perceapă* mediul (prin intermediul unor senzori) şi sunt capabili să răspundă schimbărilor care apar în acesta;
- *activitate*: agenţii nu au doar capacitatea de a acţiona ca răspuns la schimbarea mediului, ei sunt capabili de a avea un comportament "orientat spre scop", preluând iniţiativa;
- *abilitate socială*: agenţii interacţionează cu alţi agenţi prin intermediul unui *limbaj de comunicare a agenţilor*.

Chiar dacă domeniul agenţilor inteligenţi provine în mare măsură din cercetări în domeniul Inteligenţei Artificiale, singura cerinţă de inteligenţă care se face, în general, asupra agenţilor, este ca aceştia să ia decizii "acceptabile" despre ce acţiuni să efectueze în mediul lor, astfel ca aceste acţiuni să fie utile. Alte cerinţe legate de inteligenţă vor fi determinate de domeniul în care agentul este aplicat: spre exemplu, nu toţi agenţii trebuie să aibă capacitatea de a învăţa.

În consecinţă, aplicarea şi exploatarea tehnologiei bazate pe agenţi poate fi privită, în esenţă, ca o problemă de "ştiinţa calculatoarelor". Agenţii sunt simple componente soft care trebuie proiectate şi implementate în mare parte în acelaşi mod ca şi celelalte componente soft. Totuşi, tehnicile de Inteligenţă Artificială *sunt* deseori cele mai potrivite căi pentru a construi agenţi.

8.1.2 Specificarea sistemelor bazate pe agenţi

Un mediu de specificare a agenţilor trebuie să fie capabil să pună în evidenţă cel puţin următoarele aspecte ale unui sistem bazat pe agenţi:

- *opiniile* pe care le are agentul despre lume;
- *interacţiunea viitoare* pe care agentul o are cu mediul în care se află;
- *scopurile* pe care agentul şi le propune;
- *acţiunile* pe care agentul le efectuează precum şi efectele acestor acţiuni.

Pentru reprezentarea acestor aspecte ale unui sistem bazat pe agenţi, o posibilă abordare ar fi *logica temporală modală*. Un mediu de specificare bazat pe logica temporală modală va conţine:

- conective logice modale pentru reprezentarea opiniilor agentului;
- conective temporale logice pentru reprezentarea dinamicii sistemului - comportamentul său în viitor;
- conective logice modale pentru reprezentarea dorinţelor, intenţiilor, obligaţiilor agentului;
- un mecanism de reprezentare a acţiunilor pe care agentul de efectuează.

8.2 Metode formale în Inteligenţa Artificială Distribuită. Consideraţii generale

Este un lucru foarte clar că în ultima vreme aplicaţiile bazate pe agenţi devin din ce în ce mai importante. Agenţii sunt proiectaţi pentru medii extrem de complexe, unde eşecul sau comportamentul necorespunzător al unui agent poate cauza pierderi importante. De aceea, una din marile provocări ale domeniului este dezvoltarea de tehnici care să asigure faptul că agenţii se vor comporta aşa cum ne aşteptăm să o facă (aşa cum

au fost proiectaţi) - sau, în cel mai rău caz nu se vor comporta în moduri inacceptabile sau nedorite.

Bineînţeles, asigurarea corectitudinii este o provocare pentru toate domeniile informaticii. Studii anterioare în domeniu au arătat că metodele formale pot fi considerate o bază solidă pentru a crea sisteme cu erori minimale.

Agenţii sunt potriviţi pentru realizarea unor astfel de sisteme, datorită faptului că furnizează abstractizări de nivel înalt pentru sistemele complexe. Aceste abstractizări pot conduce de fapt la tehnici simple pentru proiectare şi dezvoltare, deoarece oferă o abordare care evită complexitatea inerentă în aplicaţiile de dimensiuni mari.

Metodele formale în IA, şi nu numai, oferă înţelegerea sistemelor care sunt proiectate la un nivel mai înalt decât implementarea lor specifică. Acestea pot furniza un mod de a asista specificaţiile de depanare şi a valida implementările sistemului în conformitate cu specificaţiile sale precise.

Oricum, rolul metodelor formale în IA (ca şi în alte domenii ale ştiinţei calculatoarelor) este destul de controversat. În ciuda avantajelor menţionate anterior, mulţi practicieni consideră că metodele formale nu le oferă o asistenţă în eforturile lor. Acest aspect este adevărat în multe situaţii. Metodele formale, datorită preciziei lor, împiedică abordările *ad-hoc* în construirea sistemului, abordări care sunt destul de eficiente pentru sistemele cu un ciclu de viaţă scurt.

Deşi există o serie de formalisme puternice, găsirea formalismului potrivit pentru un anumit sistem este o adevărată provocare. Un astfel de formalism ar oferi un nivel de expresivitate care ar fi suficient pentru problemele practice, dar nu ar fi foarte flexibil.

În ciuda diverselor controverse, există o acceptare generală a faptului că metodele formale sunt eficiente pentru sistemele având un ciclu de viaţă lung şi contribuie la dezvoltarea unei înţelegeri clare a problemelor şi soluţiilor.

De-a lungul anilor, o serie de tehnici formale dezvoltate în IA au fost utilizate cu succes în realizarea practică a unor sisteme. Aceste metode, în general nu dezvoltă întreg sistemul, ci oferă un mod de accesare a funcţionalităţii sale.

În secţiunile următoare vor fi analizate abordările cele mai importante ale metodelor formale pentru descrierea şi raţionamentul despre agenţi şi comportamentul acestora. Se va pune accentul pe modul în care aceste metode ar putea fi realizate în sisteme practice.

8.3 Fundamente logice în teoria sistemelor multiagent

În general, formalizarea sistemelor de tip agent poate fi folosită în două scopuri distincte:

- ca limbaje de specificare internă, folosite de agent (agenţi) în raţionamentul său (lor) sau în acţiunea sa (lor);
- ca metalimbaje externe utilizate de proiectant pentru specificarea, proiectarea şi verificarea anumitor proprietăţi comportamentale ale agentului (agenţilor) situat (situaţi) în medii dinamice.

Prima clasă de abordări este mai tradiţională în IA şi presupune că agenţii au capacitatea explicită de a raţiona. Astfel de agenţi sunt cunoscuţi sub numele de *cognitivi, raţionali*.

Cea de-a doua clasă de abordări este mai recentă în studiul agenţilor, deşi este mai tradiţională în restul ştiinţei calculatoarelor. Este vorba de a utiliza formalismul pentru a permite proiectantului să raţioneze despre agent. Agentul ar putea sau nu să fie capabil să raţioneze singur în momentul în care este situat în mediu.

Deşi bazele conceptuale ale celor două abordări sunt total diferite, fundamentele matematice nu sunt întotdeauna diferite. Acesta este motivul pentru care majoritatea ideilor vor fi prezentate sub forma raţionamentului necesar şi a modului în care acesta ar trebui efectuat şi doar în al doilea rând se va trata modul de realizare a raţionamentului, privit ca o componentă a agentului, sau ca un instrument pentru proiectantul acestuia. Ar fi ideal ca acelaşi limbaj logic să servească ambelor scopuri menţionate anterior.

Oricum, compromisul între expresivitate şi calculabilitate face ca acest ideal să fie, în general nerealizabil. Constrângerile în timp real impuse agenţilor situaţi în medii dinamice necesită ca limbajul intern să fie eficient din punct de vedere computaţional, în timp ce varietatea comportamentelor complexe care sunt posibile într-un sistem de agenţi autonomi distribuiţi necesită ca limbajul extern să fie cât mai expresiv.

Vom prezenta în continuare formalizările agenţilor distribuiţi din punctul de vedere al proiectantului acestora. Aceste formalizări sunt utile în realizarea sistemelor practice, facându-le să devină mai fezabile în vederea execuţiei directe de către agenţi distribuiţi.

8.3.1 Concepte de bază

Tehnicile utilizate în formalizarea IA utilizează vaste cunoştinţe din teoriile următoarelor logici: propoziţionale, modale, temporale şi dinamice. În următoarele secţiuni, vom prezenta câteva aspecte de bază ale acestor logici, care au fost utilizate pentru a furniza semantica programelor concurente.

În fapt sunt trei aspecte importante ale unei logici:

- *formulele bine definite* (*well-formed formulas*) ale unei logici sunt aserţiunile (afirmaţiile) care pot fi făcute în logica respectivă; acestea sunt specificate sub forma unui limbaj formal care stă la baza logicii date;
- *teoria demostraţiei* (*proof-theory*) care include axiomele şi regulile de inferenţă, care stabilesc relaţii între formulele bine definite;
- *teoria modelului* (*model-theory*) care furnizează semnificaţiile formale ale formulelor bine definite.

Limbajul şi teoria demonstraţiei formează *sintaxa*; teoria modelului se numeşte *semantică*. Scopul semanticii este de a asocia formulele unor reprezentări (versiuni) simplificate ale realităţii care ne interesează, aşa numitul *model*.

Deşi logica clasică abordează doar adevărul respectiv falsitatea unei formule, deoarece modelele sunt deseori mari şi structurate, deseori avem nevoie să specificăm o componentă a modelului, în raport cu care adevărul sau falsitatea unei formule ar implica semnificaţia intuitivă pe care încercăm să o formalizăm. Se va folosi termenul *indice* pentru a face referire la o astfel de componentă, o parte a lumii (mediului), o locaţie spaţială, un moment sau o perioadă de timp.

O formulă este *satisfăcută* într-un model, pentru un anumit indice al acestuia, dacă şi numai dacă formula este adevărată în acea componentă. Pentru un model *M*, indice *i* şi o formulă *p*, acest lucru se notează prin $M \models_i p$. O formulă este *validă* într-un

model *M* dacă şi numai dacă este satisfăcută pentru toţi indicii modelului; acest lucru se notează prin $M \models p$.

Vom prezenta în continuare o serie de limbaje formale care tratează concepte ca adevăr, posibilitate, acţiune, timp, opinie, cerere, intenţie. Acestor limbaje li se va specifica sintaxa şi un set de reguli informale. Vom conveni în cele ce urmează ca regulile sintactice să le prefixăm cu *SIN,* iar regulile semantice cu *SEM.*

8.3.2 Logica propoziţională şi predicativă

Logica propoziţională este cea mai simplă şi una din cele mai des utilizate logici pentru reprezentarea informaţiilor factuale, deseori legate de mediile agenţilor. Formulele în această logică sunt construite din

- *propoziţii atomice*, care exprimă, în mod intuitiv, fapte despre mediu (lume);
- *conective funcţionale*: $\wedge, \vee, \neg, \rightarrow$.

L_P este limbajul logicii propoziţionale şi este definit de următoarele reguli (vom presupune că Φ este o mulţime dată de propoziţii atomice):

$SIN-1$ $\psi \in \Phi$ implică $\psi \in L_P$

$SIN-2$ $p, q \in L_P$ implică $p \wedge q, \neg p \in L_P$

Fie $M_o \overset{def}{=} < L >$ modelul formal pentru L_P. $L \subseteq \Phi$ este o *interpretare* sau *etichetă. L* identifică mulţimea propoziţiilor atomice care sunt adevărate.

Ca urmare, semnificaţia formulelor care nu sunt atomice poate fi definită recursiv:

$SEM-1$ $M_0 \models \psi$ *ddaca* $\psi \in L$, unde $\psi \in \Phi$

$SEM-2$ $M_0 \models p \wedge q$ *ddaca* $M_0 \models p$ *si* $M_0 \models q$

$SEM-3$ $M_0 \models \neg p$ *ddaca* $M_0 \not\models p$

Propoziţiile atomice şi combinaţiile booleene ale acestora sunt folosite pentru descrierea stărilor sistemului, dar acestea nu ilustrează modul în care sistemul evoluează sau a evoluat.

8.3.3 Logica modală

Logica modală este foarte des utilizată în Inteligenţa Artificială pentru a face referire la alt tip de semnificaţie a formulelor, decât cel utilizat în logica propoziţională clasică.

În forma ei generală, logica modală a fost utilizată de filozofi pentru a investiga diferite "*variante*" ale adevărului, cum ar fi "*posibil adevărat*" şi "*în mod necesar adevărat*".

În studiul agenţilor, logica modală este folosită pentru a da semnificaţia unor concepte ca opinie şi cunoştinţă. În limbajele modale, logica propoziţională clasică este extinsă cu doi *operatori modali*: $\Diamond$ (pentru posibilitate) şi $\Box$ (pentru necesitate).

Limbajul modal L_M este definit după cum urmează:

$SIN-3$ regulile pentru L_P (cu L_P înlocuit cu L_M)

$SIN-4$ $p \in L_P$ implică $\Diamond p$ şi $\Box p \in L_M$

Modelele pentru logica modală necesită structuri adiţionale înafară de M_0. Semantica logicii modale este dată, în mod tradiţional, sub forma unor mulţimi de expresii ale unei aşa numite *lumi(medii) posibile*.

O lume (mediu) poate fi interpretată în mai multe moduri:

- ca o posibilă stare a lucrurilor, aproximativ corespunzătoare unei interpretări, ca în semantica L_P;
- ca un istoric, adică o secvenţă de stări ale lucrurilor;
- ca o mulţime a posibilelor istorice pornind dintr-o stare dată.

Ultimele două puncte de vedere sunt mai uzuale în literatura filozofică. Noi vom considera lumea (tehnic vorbind) ca o stare a lucrurilor şi câteodată corespunzând unui posibil istoric.

Considerând ca primitive o mulţime de lumi, structura modelului se va obţine realizând o corespondenţă între diferite lumi prin intermediul unei relaţii binare de *accesibilitate*. Intuitiv, această relaţie ne indică faptul că lumile sunt în interiorul unei zone de posibilitate, din punctul de vedere al unei lumi date. O condiţie e *posibilă*, dacă e adevărată undeva în interiorul zonei de posibilitate; o condiţie e *necesară* dacă este adevărată peste tot în interiorul zonei de pobilitate.

Fie $M_1 \overset{def}{=} < W, L, R >$, unde W este o mulţime de lumi, $L : W \to 2^{\Phi}$ este o funcţie care pentru o lume furnizează mulţimile de formule adevărate în acea lume, iar $R \subseteq WxW$ este o relaţie de accesibilitate. Odată ce modelul este construit, indicele semnificativ este lumea posibilă în raport cu care se evaluează formula.

$SEM-4 \quad M_1 \models_w \psi \; ddaca \; \psi \in L(w)$, unde $\psi \in \Phi$

$SEM-5 \quad M_1 \models_w p \wedge q \; ddaca \; M_1 \models_w p \; si \; M_1 \models_w q$

$SEM-6 \quad M_1 \models_w \neg p \; ddaca \; M_1 \not\models_w p$

$SEM-7 \quad M_1 \models_w \Diamond p \; ddaca \; (\exists w' : R(w,w') \; \& \; M_1 \models_{w'} p)$

$SEM-8 \quad M_1 \models_w \Box p \; ddaca \; (\forall w' : R(w,w') \Rightarrow M_1 \models_{w'} p)$

Exemplu

Logica modală ne permite să reprezentăm *condiţionări stricte*, care oferă o formalizare mai precisă a implicaţiilor din limbajul natural, decât cea oferită de operatorii propoziţionali. Formula $\Box(p \to q)$ este adevărată nu numai dacă p este *fals*, ci şi dacă p şi q sunt în mod corespunzător conectate la toate lumile posibile.

Important este faptul că proprietăţile algebrice ale relaţiei de accesibilitate se traduc în proprietăţi ataşate logicii. Câteva proprietăţi algebrice uzuale sunt următoarele:

- R este *reflexivă* ddacă $(\forall w : (w,w) \in R)$
- R este *serială* ddacă $(\exists w' : (w,w') \in R)$
- R este *tranzitivă* ddacă
  $(\forall w_1, w_2, w_3 : (w_1, w_2) \in R \; \& \; (w_2, w_3) \in R \Rightarrow (w_1, w_3) \in R)$
- R este *simetrică* ddacă $(\forall w_1, w_2 : (w_1, w_2) \in R \Rightarrow (w_2, w_1) \in R)$
- R este *euclidiană* ddacă
  $(\forall w_1, w_2, w_3 : (w_1, w_2) \in R \; \& \; (w_1, w_3) \in R \Rightarrow (w_2, w_3) \in R)$

Se poate arăta faptul că un model care satisface proprietăţile anterioare, validează şi următoarele formule:

- $\Box p \rightarrow p$
- $\Box p \rightarrow \Diamond p$
- $\Box p \rightarrow \Box\Box p$
- $p \rightarrow \Box\Diamond p$
- $\Diamond p \rightarrow \Box\Diamond p$

Deoarece formulele anterioare nu depind de p, ele sunt privite ca nişte *scheme* care pot fi aplicate în orice situaţie. În literatura informatică, sunt cunoscute ca schemele *T, D,* 4, *B* şi 5.

8.3.4 Logica deontică

Logica deontică se referă la ce ar trebui să fie sau la ce un agent este *obligat* să facă. Logica deontică tradiţională introduce un operator **Obl** pentru *obligat*, al cărui operator dual este **Per** pentru *permis*.

Logica deontică poate fi specificată ca o logică modală, având ca axiomă principală următoarea:

Obl $p \rightarrow \neg$ **Obl** $\neg p$,

ceea ce înseamnă că agentul este obligat să determine *p* numai dacă el nu este obligat să determine $\neg p$. Restul logicii este uşor de înţeles.

8.3.5 Logica dinamică

Logica dinamică poate fi interpretată ca logica modală a acţiunii. Spre deosebire de logica modală tradiţională, totuşi, operatorii de necesitate şi posibilitate se bazează pe varietatea de acţiuni disponibile. Ca o consecinţă a acestei flexibilităţi, logica modală şi-a găsit utilizare în anumite domenii ale IAD.

Vom considera în continuare logica dinamică propoziţională a programelor regulare, care este cea mai utilizată variantă. Această logică are un sublimbaj bazat pe expresii regulare utilizate pentru definirea expresiilor de acţiune - aceste acţiuni corespund programelor Algol-60, de unde şi numele de *programe regulare*. Vom defini L_D împreună cu L_R ca o definiţie auxiliară.

Pentru definirea limbajului bazat pe logica dinamică, vom considera o mulţime simboluri ale acţiunilor atomice, notată B .

$SIN-5$ regulile de la L_P, aplicate pentru L_D

$SIN-6$ $\beta \in \mathrm{B}$ implică $\beta \in L_R$

$SIN-7$ $a,b \in L_R$ implică $a;b,(a+b),a^* \in L_R$

$SIN-8$ $p \in L_D$ implică $p? \in L_R$

$SIN-9$ $a \in L_R$ şi $p \in L_D$ implică $[a]p$ şi $< a > p \in L_R$

Intuitiv, acţiunile atomice sunt cele pe care agentul le poate executa direct. Programul $a;b$ înseamnă executarea a şi b în secvenţă. Programul $a+b$ înseamnă executarea a sau b, oricare este posibil a fi executat. Aceasta este o alegere nedeterministă - totuşi, un program nedeterminist ar putea să nu fie executabil practic,

deoarece necesită o căutare opţională pentru a deduce care ramură să fie într-adevăr aleasă. Programul $p?$ este o acţiune bazată pe confirmarea valorii de adevăr a propoziţiei p. Dacă p este adevărată, acţiunea reuşeste, fără a afecta starea lumii. Dacă p este falsă, acţiunea eşuează, iar ramura acţiunii a cărei parte este p se termină cu eşec - e ca şi cum ramura nu ar exista de fapt. Programul $a*$ înseamnă 0 sau mai multe (dar număr finit) iteraţii ale lui a.

Exemplu

Programul Algol-60 **if *q* then *a* else *b* endif** s-ar traduce ca $((q?:a)+((\neg q)?:b))$.
Dacă q este realizabilă, atunci ramura $(\neg q)?$ eşuează, deci a va trebui executat. În caz contrar, b va trebui executat.

Semantica logicii dinamice va fi dată în legătură cu un model care include o mulţime de stări (sau lumi) conectate prin posibile tranziţii bazate pe acţiunile din B.

Fie $M_2 \stackrel{def}{=} < W, L, \delta >$, W şi L având aceeaşi semnificaţia ca şi în cazul logicii modale, iar $\delta \subseteq WxBxW$ este o relaţie de tranziţie. Este util să fie definită o clasă de relaţii de accesibilitate bazate pe L_R.

$PR-1$ $R_\beta(w,w')$ ddacă $\delta(w,\beta,w')$

$PR-2$ $R_{a;b}(w,w')$ ddacă $(\exists w'': R_a(w,w'') \& R_b(w'',w'))$

$PR-3$ $R_{a+b}(w,w')$ ddacă $R_a(w,w')$ *sau* $R_b(w,w')$

$PR-4$ $R_{a*}(w,w')$ ddacă

$(\exists w_0,...,w_n : (w=w_0) \& (w'=w_n) \& (\forall i \in [1,n) \Rightarrow R_a(w_i,w_{i+1})))$

$SEM-9$ $M_2 \models_w < a > p$ ddacă $(\exists w': R_a(w,w') \& M_2 \models_{w'} p)$

$SEM-10$ $M_2 \models_w [a]p$ ddacă $(\forall w': R_a(w,w') \Rightarrow M_2 \models_{w'} p)$

8.3.6 Logica temporală

Logica temporală este, bineînţeles, logica timpului. Există câteva variante ale acestei logici. Dintre acestea, cele mai importante sunt următoarele:

- *liniară - neliniară (ramificată)*: după cum timpul este sau nu privit sub forma unei singure direcţii de evoluţie temporală. Ramificarea (neliniaritatea) ar putea fi în trecut, în viitor, sau în ambele;
- *discretă - densă*: după cum timpul este văzut sub forma unor paşi discreţi (ca şi mulţimea numerelor naturale) sau ca întotdeauna având stări intermediare (ca şi mulţimea numerelor raţionale sau reale);
- *bazată pe moment - bazată pe perioadă*: după cum particulele (atomii) timpului sunt puncte sau intervale.

Logica temporală poate fi privită ca o extensie a logicii clasice cu diferite modalităţi de reprezentare a aspectelor temporale ale formulelor logice. Logica temporală şi-a găsit aplicaţie într-un limbaj de programare a agenţilor, limbajul **M**ETATE**M** Concurent, folosit pentru specificarea comportamentelor agenţilor, şi în special pentru modelarea sistemelor reactive.

Logica temporală utilizată se bazează pe un model temporal discret şi liniar. Astfel, acesta este modelat ca o secvenţă infinită de stări discrete, cu un punct de plecare cunoscut, numit *moment* (*timp*) *iniţial*. Formulele clasice sunt utilizate pentru a reprezenta limitări (constrângeri) interne ale stărilor individuale, pe când formulele temporale reprezintă limitări între stări.

Deoarece formulele sunt interpretate ca stări particulare ale secvenţei, sunt necesari operatori care să facă referinţă la trecut şi viitor.

Operatorii temporali ce se referă la *viitor* sunt:

- operatorul '$\Diamond$' - *la un moment dat în viitor*
  $\Diamond\varphi$ este validă acum dacă φ este validă *la un moment dat* în viitor;
- operatorul '$\Box$'- *tot timpul în viitor*
  $\Box\varphi$ este validă acum dacă φ este validă *tot timpul* în viitor;

În mod similar, există operatori care permit referinţe la *trecut*.

- operatorul '*S*' - *din acel moment*
  $\varphi S \psi$ este validă acum dacă ψ a fost validă în trecut şi φ a fost validă din acel moment până în momentul prezent (exclusiv);
- operatorul '*start*' - *la început*
  start este valid doar la început (la momentul iniţial);
- operatorul $\otimes$ - *ultima oară*
  $\otimes\varphi$ este validă dacă a fost un ultim moment de timp în care φ a fost validă.

Sunt mulţi alţi operatori utilizaţi în logica temporală în general, şi în limbajul **M**ETATE**M**, în particular. Ceea ce ar fi interesant de remarcat este faptul că utilizarea logicii temporale ca bază a regulilor computaţionale furnizează un nivel suplimentar de putere expresivă în comparaţie cu cea furnizată de logica clasică. În particular, operatori ca '$\Diamond$' ne dau posibilitatea de a specifica nedeterminări temporale (în viitor).

Deşi există avantaje ale tuturor celor trei variante de logici temporale descrise anterior, vom considera modelele discrete, bazate pe moment şi cu trecut liniar, dar vom considera atât viitorul liniar cât şi cel neliniar.

Înainte de a începe formalizarea modelului temporal, să considerăm o analiză informală a timpului. Această analiză se bazează pe o mulţime de *momente*, între care există o relaţie de ordine parţială, "<", care reflectă precedenţa temporală. Fiecare moment de timp este asociat unei stări a lumii, identificată prin condiţii atomice sau propoziţii care sunt adevărate în acel moment. Un *drum* la un moment dat se defineşte ca fiind mulţimea maximală (a momentelor) ce conţine momentul dat şi toate momentele din viitor, de-a lungul unei ramuri particulare a relaţiei "<". Astfel, un drum poate fi considerat o posibilă cale de derulare a eveimentelor.

În teoria agenţilor, pentru înţelegerea intuiţiilor despre posibilităţile şi abilităţile agenţilor este importantă identificarea unuia dintre drumurile începând dintr-un moment ca fiind cel *real*. Acesta este drumul pe care lumea evoluează, presupunând că a fost în

starea indicată de momentul dat. Constrângerile despre ce ar trebui sau ce se va întâmpla, pot fi în mod natural formulate relativ la drumul real.

Figura 8.1 ilustrează schematic această analiză a timpului.

Exemplu

În Figura 8.1 sunt prezentate acţiunile a doi agenţi. Fiecare agent influenţează viitorul prin acţiunile sale, dar rezultatele depind şi de alte evenimente. Spre exemplul agentul 1 poate limita viitorul, executând una din acţiunile *a* sau *b*. Dacă alege acţiunea *a*,

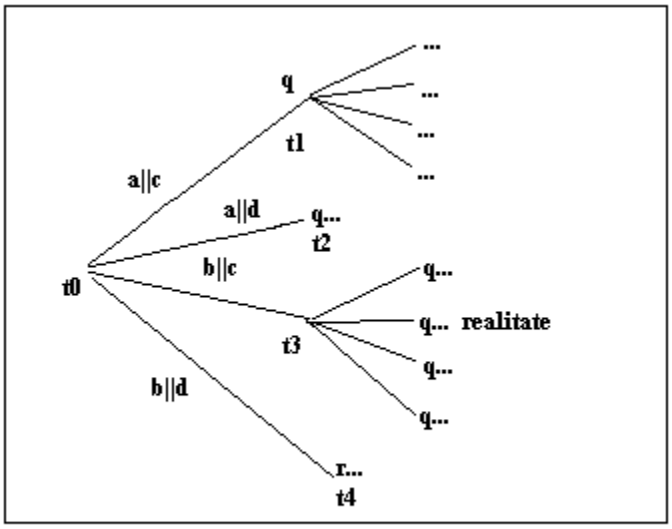

Figura 8.1. Un exemplu de structură temporală

atunci lumea evoluează de-a lungul uneia din cele două ramuri de deasupra lui t_0; dacă alege acţiunea *b*, atunci lumea evoluează de-a lungul uneia din cele două ramuri de sub t_0.

Exemplu

Tot în Figura 8.1, agentul 1 poate alege între t_1 şi t_2, pe de o parte şi t_3 şi t_4, pe de altă parte. Totuşi, agentul nu va putea alege nici între t_1 şi t_2, nici între t_3 şi t_4.

8.3.6.1 Logica temporală liniară

L_L este un limbaj temporal liniar în raport cu timpul.

$SIN-10$ regulile pentru L_P

$SIN-11$ $p,q \in L_L$ implică $p \cup q, \mathrm{X}p, \mathrm{P}p \in L_L$

$p \cup q$ este validă la momentul *t* pe un anumit drum, dacă şi numai dacă *q* va fi validă la un moment viitor pe drumul dat, iar *p* este validă în toate momentele dintre *t* şi momentul de realizare a lui *q*. F *p* înseamnă că *p* va fi validă la un moment dat în viitor pe drumul dat, şi este o prescurtare pentru $true \cup p$. G *p* înseamnă că *p* va fi tot timpul

adevărat în viitor pe drumul dat, şi este o prescurtare pentru $\neg \mathrm{F} \neg p$. $\mathrm{X}p$ înseamnă că p va fi adevărat în momentul imediat următor, iar $\mathrm{P}q$ înseamnă că q este adevărată în momentul actual.

Semantica logicii temporale liniare este dată în raport cu un model $M_3 \overset{def}{=} < \mathrm{T}, <, [] >$, unde T reprezintă o mulţime de momente, "<" este relaţia de ordine temporală, iar "[]" furnizează semnificaţia propoziţiilor atomice. "[]" este de fapt echivalentul unei interpretări $L : w \in [\psi]$ ddacă $\psi \in L(w)$.

$SEM-11$ $M_3 \models_t \mathrm{P}_p$ ddacă $(\exists t' : t' < t$ şi $M_3 \models_{t'} p)$

$SEM-12$ $M_3 \models_t \mathrm{X}_p$ ddacă $M_3 \models_{t+1} p$

$SEM-13$ $M_3 \models_t p \cup q$

ddacă $(\exists t' : t \le t'$ şi $M_3 \models_{t'} q$ şi $(\forall t'' : t \le t'' \le t' \Rightarrow M_3 \models_{t''} p))$

Pentru dezvoltările formale ulterioare, este util de reţinut faptul că M_3 este liniar, astfel încât "<" este aici o relaţie de ordine totală.

8.3.6.2 Logica temporală neliniară bazată pe acţiuni

L_B este un limbaj temporal neliniar bazat pe acţiune.

Este superior limbajelor L_L şi L_D, tratând proprietăţile esenţiale ale acţiunilor şi ale timpului, care sunt deosebit de importante în specificarea agenţilor.

Din punct de vedere formal, L este mulţimea minimală definită de regulile prezentate mai jos. Cu L_s se va nota o mulţime de "formule referitoare la drumuri", utilizate ca o definţie auxiliară. X este o mulţime de variabile, iar A este o mulţime de simboluri ale agenţilor. Vom da semnificaţii intuitive ale construcţiilor acestui limbaj formal, urmând definiţiile sintactice de mai jos.

$SIN-12$ regulile din L_P

$SIN-13$ $p, q \in L_B$ implică $\mathrm{P}p, (\vee a : p) \in L_B$

$SIN-14$ $L_B \subseteq L_s$

$SIN-15$ $p, q \in L_s$, $x \in A$ şi $a \in B$ implică

$p \wedge q, \neg p, p \cup q, \mathrm{X}p, x[a]p, x < a > p \in L_s$

$SIN-16$ $p \in L_s$ implică $\mathrm{A}p, \mathrm{R}\, p \in L_B$

$SIN-17$ $p \in (L_s \setminus L_B)$ şi $a \in X$ implică $(\vee a : p) \in L_s$

Formulele din L_B se referă la momente; cele din L_s se referă la drumuri ca şi în modelele L_L. Deşi $L_B \subseteq L_s$, formulele din L_B au o semantică unică.

Operatorul de ramificare temporală, A semnifică "*pe toate drumurile la momentul prezent*". "*Momentul prezent*" se referă la momentul la care o anumită formulă

este evaluată. O notaţie prescurtată ar fi operatorul E, cu semnificaţia "*pe anumite drumuri la momentul prezent*". Cu alte cuvinte, $\mathrm{E}\, p \equiv \neg \mathrm{A} \neg p$.

Exemplu

În Figura 8.1, $\mathrm{EF}\, r$ şi $\mathrm{AF}(q \vee r)$ sunt valide pentru t_0, pentru că r e validă la un anumit moment pe un anumit drum din t_0, iar q este validă la un anumit moment pe orice drum.

Operatorul de *realitate*, R, semnifică "pe drumul *real*, la momentul prezent". R nu este inclus în logica temporală tradiţională, dar aici ajută la conectarea intuiţiilor despre ce ar trebui şi ce se va întâmpla.

Exemplu

În Figura 8.1, $\mathrm{RF}\, q$ este validă la momentul t_0, atâta timp cât q este validă la un moment dat pe drumul real identificat la t_0.

L_B conţine, deasemenea, operatori asupra acţiunilor. Aceştia sunt adaptaţi şi generalizaţi din L_D, în care operatorii de acţiune furnizează formule referitoare la stări, în timp ce în L_B aceştia furnizează formule referitoare la drumuri. Operatorii din L_B includ operatorii din L_D.

$x[a]p$ este validă pe un anumit drum S la un moment t, dacă şi numai dacă, dacă x execută a de-a lungul lui S, începând cu momentul t, atunci p este validă pe drumul S, la momentul în care a se termină.

$x < a > p$ este validă pe un anumit drum S la un moment t, dacă şi numai dacă x execută a de-a lungul lui S, începând cu momentul t, iar p este validă la momentul în care a se termină.

Exemplu

În Figura 8.1, $\mathrm{E} < b > r$ şi $\mathrm{A}[a]q$ sunt valide la t_0, deoarece r este validă la sfârşitul lui b, pe un anumit drum, iar q este validă la sfârşitul lui a pe fiecare drum. În mod similar, $\mathrm{A}[d](q \vee r)$ este validă la t_0, iar $\mathrm{A}[e]true$ este validă la t_0, deoarece e nu nu poate fi executată la momentul t_0.

Notaţia $(\vee a : p)$ înseamnă că există o acţiune la sfârşitul execuţiei căreia p devine adevărată.

Exemplu

În Figura 8.1, $(\vee e : \mathrm{E}\, x < e > true \wedge \mathrm{A}x[e]q)$ e validă la t_0. Ceea ce înseamnă că există o acţiune, să spunem a, astfel încât x execută a pe un anumit drum începând cu momentul t_0 şi pe toate drumurile pe care a este executat, rezultă q va fi adevărat. Cu alte cuvinte, există o acţiune posibilă care va conduce totdeauna la q. Această paradigmă este folosită în formalizarea "know-how".

Fie $M_4 \stackrel{def}{=} <\mathrm{T},<,[],\mathrm{R}>$ un model formal pentru L_B. Spre deosebire de M_3, M_4 este ramificat (neliniar), iar "[]" se aplică deasemenea acţiunilor. Cu alte cuvinte, relaţia "<" este ramificată. S-ar putea partiţiona T într-un număr de componente conectate, fiecare dintre acestea ar corespunde lumilor, în interpretarea lor tradiţională. Pentru o propoziţie atomică p, $[p]$ este mulţimea momentelor la care p este validă; pentru acţiunea a a unui agent x, $[a]^x$ este mulţimea intervalelor în care a este executat de x. Aceste intervale sunt notate $[S;t,t']$, astfel încât a începe la t şi se termină la t', unde $t' \in S$. R alege la fiecare moment drumul *real* la acel moment.

Pentru simplificare, se va presupune că fiecare simbol de acţiune este cuantificat cel mult o dată în orice formulă. $p|_b^a$ este formula rezultată prin substituirea tuturor apariţiilor lui a în p cu b. Se va presupune, deasemenea, că simbolul unui agent corespunde unui agent unic pretutindeni în model.

Formal, avem următoarele reguli semantice:

$SEM-14$ $M_4 \models_t \psi$ ddacă $t \in [\psi]$, unde $\psi \in \Phi$

$SEM-15$ $M_4 \models_t p \wedge q$ ddacă $M_4 \models_t p$ şi $M_4 \models_t q$

$SEM-16$ $M_4 \models_t \neg p$ ddacă $M_4 \not\models_t p$

$SEM-17$ $M_4 \models_t \mathrm{A}p$ ddacă $(\forall S : S \in S_t \Rightarrow M_4 \models_{S,t} p)$

$SEM-18$ $M_4 \models_t \mathrm{R}\, p$ ddacă $M_4 \models_{R(t),t} p$

$SEM-19$ $M_4 \models_t \mathrm{P}p$ ddacă $(\exists t' : t' < t$ şi $M_4 \models_{t'} p)$

$SEM-20$ $M_4 \models_{S,t} \mathrm{X}p$ ddacă $M_4 \models_{S,t+1} p$

$SEM-21$ $M_4 \models_t (\forall a : p)$ ddacă $(\exists b : b \in \mathrm{B}$ şi $M_4 \models_t p|_b^a)$, unde $p \in L$

$SEM-22$ $M_4 \models_{S,t} (\forall a : p)$ ddacă $(\exists b : b \in \mathrm{B}$ şi $M_4 \models_{S,t} p|_b^a)$, unde $p \in (L_s \setminus L)$

$SEM-23$ $M_4 \models_{S,t} p \bigcup q$ ddacă $(\exists t' : t < t'$ şi $M_4 \models_{S,t'} q$ şi $(\forall t'' : t \le t'' \le t' \Rightarrow M_4 \models_{S,t''} p))$

$SEM-24$ $M_4 \models_{S,t} x[a]p$ ddacă $(\forall t' \in S : [S;t,t'] \in [a]^x \Rightarrow M_4 \models_{S,t'} p)$

$SEM-25$ $M_4 \models_{S,t} x < a > p$ ddacă $(\exists t' \in S : [S;t,t'] \in [a]^x \,\&\, M_4 \models_{S,t'} p)$

$SEM-26$ $M_4 \models_{S,t} p \wedge q$ ddacă $M_4 \models_{S,t} p$ şi $M_4 \models_{S,t} q$

$SEM-27$ $M_4 \models_{S,t} \neg p$ ddacă $M_4 \not\models_{S,t} p$

$SEM-28$ $M_4 \models_{S,t} p$ ddacă $M_4 \models_t p$, unde $p \in L$

8.3.7 Logici nemonotone

O **logică nemonotonă** este o logică formală în care procesul de deducţie nu este **monoton**. Cele mai multe logici formale folosesc raţionament monoton, ceea ce înseamnă că adăugarea unei formule teoriei nu va produce niciodată o reducere a mulţimii consecinţelor. Din punct de vedere intuitiv, monotonicitatea presupune că învăţarea unei noi piese de cunoaştere nu va reduce mulţimea a ceea ce se cunoaşte (a fi

adevărat). O logică monotonică nu poate rezolva diferite sarcini de raţionament cum ar fi: ***raţionamentul prin default-uri*** (***default reasoning***) (fapte care pot fi cunoscute doar datorită faptului că nu se poate dovedi contrariul lor), ***raţionamentul prin abducţie*** (***abductive reasoning***) (faptele sunt deduse doar ca şi explicaţii probabile), ***raţionamentul despre cunoştinte*** (***reasoning about knowledge***) (necunoaşterea unui fapt va trebui retrasă atunci când faptul devine adevărat) şi ***revizuirea părerilor*** (***belief revision***) (cunoştinţe noi pot contrazice vechile păreri).

Default reasoning

Un exemplu de presupunere *default* este faptul că "Păsările zboară". Ca un rezultat al acestei afirmaţii, dacă se cunoaşte că un anumit animal este pasăre, şi nu se cunoaşte nimic altceva, se poate presupune că zboară. Acest fapt va putea fi, însă, retractat dacă mai târziu se află (învaţă) că animalul respectiv este pinguin.

Acest exemplu arată faptul că o logică care modelează raţionamentul prin *default-uri* nu trebuie să fie monotonă. Logicile care formalizează raţionamentul prin *default-uri* pot fi împărţite în două categorii: logici capabile să abordeze presupunerile *default* (***default logic, defeasible logic*** şi ***answer set programming***) şi logici care formalizează presupunerea *default* specifică cum că faptele care nu sunt cunoscute a fi adevărate pot fi presupuse implicit false (***closed word assumption*** şi ***circumscription***).

Abductive reasoning

Raţionamentul prin abducţie este procesul de deducţie al celor mai probabile explicaţii ale faptelor cunoscute. O logică abductivă nu trebuie să fie monotonă deoarece cele mai probabile explicaţii nu sunt în mod necesar corecte. Spre exemplu, cea mai probabilă explicaţie pentru că "Iarba este udă" este faptul că a plouat; însă această explicaţie va putea fi retrasă atunci când se învaţă cauza reală a faptului că iarba s-a udat, şi anume *stropitoarea*. Deoarece explicaţia veche (faptul că a plouat) este retrasă datorită adăugării unei noi piese de cunoaştere (o stropitoare a fost folosită), orice logică care modelează explicaţiile este nemonotonă.

Reasoning about knowledge

Dacă o logică include formule care indică faptul că un anumit lucru nu este cunoscut, această logică nu ar trebui să fie monotonă. Într-adevăr, învăţarea unui lucru (fapt) care iniţial nu era cunoscut conduce la o ştergere a formulei care specifica că acel lucru era necunoscut . Această modificare (o ştergere din cauza unei adăugări) violează condiţia monotonicităţii. O logică pentru raţionament despre cunoştinţe este ***logica autoepistemică*** (***autoepistemic logic***).

Belief revision

Revizuirea opiniilor (părerilor) este procesul de schimbare a părerilor pentru adaptarea la o nouă părere care ar putea fi inconsistentă la vechile păreri. În presupunerea că noua părere este corectă, anumite păreri din cele vechi vor trebui retrase pentru a menţine consistenţa. Această retragere ca urmare a unei adăugări a unei noi păreri face ca

orice logică pentru revizuirea părerilor să fie nemonotonă. Abordarea de revizuire a părerilor este alternativa ***logicilor paraconsistente*** (***paraconsistent logics***), logici care tolerează inconsistenţa mai degrabă decît să o elimine.

Nu vom intra mai mult în amănunte legate de teoria logicilor nemontone, deoarece este foarte vastă. Vrem doar să evidenţiem faptul că aceste tipuri de logice sunt foarte des folosite în raţionament în sistemele multi-agent, mai ales în situaţiile în care mediile în care acţionează agenţii sunt necunoscute şi dinamice. Deasemenea, logicile nemonotone pot fi folosite în comunicarea între agenţi în sisteme multi agent.

8.4 Primitive cognitive în formalizarea sistemelor bazate pe agenţi

În analiza sistemelor bazate pe agenţi, o deosebită importanţă o au specificaţiile cognitive ale acestora, ceea ce ar presupune analiza agenţilor la *nivelul cunoştinţelor*. Acest lucru poate fi realizat printr-o *atitudine intenţională* în direcţia agenţilor, sau privind agenţii la nivelul de *cunoaştere*.

Specificaţiile derivate din noţiuni cognitive sunt, probabil, cea mai semnificativă contribuţie a Inteligenţei Artificiale în domeniul agenţilor.

Specificaţiile cognitive se referă la concepte precum: opinii, cunoştinţe, dorinţe şi intenţii. Aceste specificaţii ne permit să definim starea curentă a unui agent, ce ar trebui să facă agentul şi cum ar trebui să se comporte în diferite situaţii, fără legătură cu modul în care agentul este implementat. Specificaţiile derivate din noţiuni cognitive sunt poate cele mai importante contribuţii ale Inteligenţei Artificiale în domeniul agenţilor.

În continuare, vom face referire la conceptele cognitive de opinie, cerere, intenţie (OCI), dându-le acestor concepte o definiţie formală. Logica definită în această secţiune poate fi folosită pentru a raţiona despre agenţi şi despre modul în care opiniile, intenţiile şi acţiunile acestora determină satisfacerea dorinţelor (cererilor).

Pentru a realiza formalizarea, se vor introduce operatorii modali **Opi** (opinie), **Cer** (cerere), $\mathbf{K_h}$ ("know-how"), **Int** (intenţie) şi $\mathbf{K_t}$ ("know-that"). Vom nota prin L_s o mulţime de formule, iar prin **A** o mulţime de simboluri pentru agenţi.

$SIN-18$: $p \in L_s$ şi $x \in \mathbf{A}$ implică $(x\ \mathbf{Int}\ p)$, $(x\ \mathbf{K_h}\ p)$, $(x\ \mathbf{K_t}\ p)$, $(x\ \mathbf{Dor}\ p) \in L_I$

Semantica pentru L_I este dată în legătură cu modelul $M_5 \stackrel{def}{=} < \mathrm{T}, <, [], \mathrm{R}, \mathbf{O}, \mathbf{C}, \mathbf{I} >$.

Exemplu

Să considerăm cazul unui agent care doreşte să câştige o loterie şi intenţionează să cumpere un bilet de loterie la un moment dat, dar nu crede că va câştiga vreodată loteria. Starea mentală a agentului ar putea fi reprezentată prin următoarea formulă:

$\mathbf{Dor}\mathrm{AF}\ castiga \wedge \mathbf{Int}\mathrm{EF}\ cumpara \wedge \neg\, \mathbf{Opi}\mathrm{AF}\ castiga$

Menţionăm că formula AFp este adevărată dacă p se va realiza la un moment dat pe toate drumurile, iar formula EFp este adevărată dacă p se va realiza la un moment dat pe un anumit drum.

8.4.1 Cunoaştere şi Opinii

Se va introduce o relaţie de *accesibilitate a opiniilor*, **O**, pentru a da semnificaţia operatorului de opinie, care se comportă ca un operator modal de necesitate, cum este operatorul modal □. **O** asignează fiecărui agent, la orice moment, mulţimea momentelor pe care agentul le consideră posibile la acel moment. Cunoaşterea ("know-that") este în mod frecvent definită ca fiind o opinie adevărată.

În mod tradiţional, pentru a modela opiniile, se presupune că operatorul **O** este serial, simetric şi euclidian. Pentru a modela cunoaşterea, se presupune în plus că operatorul este şi reflexiv. În acest caz, devine o relaţie de echivalenţă.

Dacă □ este tratat ca opinie (cunoaştere), atunci vom avea următoarele afirmaţii logice:

- dacă un agent crede o condiţie, atunci crede faptul că o crede;
- dacă un agent nu crede o condiţie, crede faptul că nu o crede.

Astfel, aceste scheme vor fi referite ca *intenţie pozitivă*, respectiv *negativă*. Intenţia negativă este o presupunere puternică pentru agenţii limitaţi. Cele două afirmaţii pot fi redate prin următoarea regulă semantică:

$$SEM-1: \quad M_5 \models_t x\,\mathbf{Opi}\,p \text{ ddacă } (\forall t': (t,t') \in \mathbf{O}(x,t) \Rightarrow M_5 \models_{t'} p)$$

O depinde de momentul actual, astfel încât agentul îşi poate schimba opiniile de-a lungul timpului.

8.4.2 Cereri şi scopuri

C asociază fiecărui moment o mulţime de momente pentru a reprezenta cererile (dorinţele) agentului. Agentul are o dorinţă ϕ la un moment dat dacă şi numai dacă ϕ este adevărată în toate lumile **C**-accesibile agentului în acel moment.

$$SEM-2: \quad M_5 \models_t x\,\mathbf{Dor}\,p \text{ ddacă } (\forall t': (t,t') \in \mathbf{C}(x,t) \Rightarrow M_5 \models_{t'} p)$$

Din punct de veder psiho-filozofic, dorinţele pot fi inconsistente şi agenţii nu trebuie să cunoască semnificaţia realizării acestor dorinţe. Dorinţele sunt intrări în procesul de deliberare al agentului şi al cărui rezultat este alegerea unei submulţimi de dorinţe care sunt atât consistente, cât şi realizabile. Aceste dorinţe consistente şi realizabile se numesc *scopuri*. Pentru simplificare, dorinţele prezentate sunt consistente din punct de vedere logic.

8.4.3 Intenţii

La fiecare moment în cadrul modelului, **I** asignează fiecărui agent o mulţime de drumuri interpretate ca fiind selectate (preferate) de către agent. Intenţiile sunt definite ca fiind *condiţiile care în mod inevitabil sunt valide pe fiecare dintre drumurile selectate*.

Definiţia formală pentru realizarea intenţiilor este următoarea:

$$SEM-3: \quad M \models_t x\,\mathbf{Int}\,p \text{ ddacă } (\forall S: S \in \mathbf{I}(x,t) \Rightarrow M \models_{S,t} \mathrm{F}\,p)$$

În regula anterioară formula Fp este validă dacă p se va realiza la un moment dat pe drumul curent.

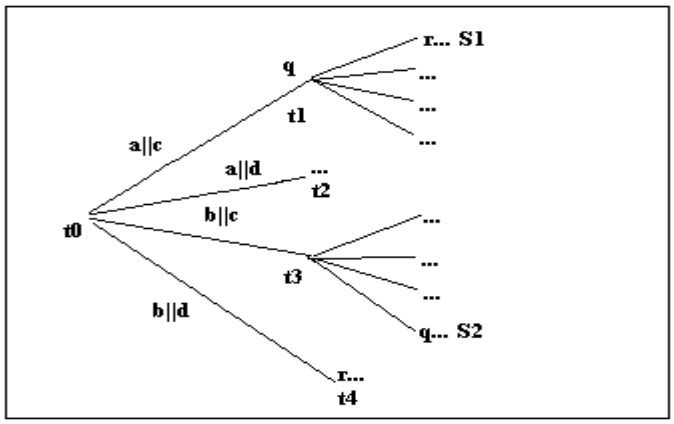

Figura 8.2. Intenţii

În Figura 8.2, în care avem reprezentaţi doi agenţi, presupunem că pe ramurile pe care nu este specificată explicit validitatea unei formule, $\neg r$ şi $\neg q$ sunt valide. Presupunem, deasemenea, că agentul x (ale cărui acţiuni sunt scrise primele pe figură) la momentul t_0 preferă drumurile S_1 şi S_2. Apoi, pe baza informaţiei informale furnizată anterior, rezultă că x are intenţia q (deoarece apare în cele din urmă pe ambele drumuri preferate) şi că nu are intenţia r (deoarece nu apare niciodată pe drumul S_2).

Definiţia anterioară validează câteva proprietăţi utile ale intenţiilor.

I1 **Satisfiabilitate:**

x **Int** $p \rightarrow$ EF p

Această proprietate are următoarea semnificaţie logică: dacă agentul x are intenţia p, atunci p va apărea în cele din urmă pe vreun drum, ceea ce înseamnă că intenţia poate fi satisfăcută. Acest lucru în general nu este valabil, deoarece mulţimea drumurilor asignate de **I** poate fi vidă. Din această cauză trebuie adăugată constrângerea ca $\mathbf{I}(x,t) \neq \phi$.

I2 **Consistenţa temporală:**

$(x$ **Int** $p \wedge x$ **Int** $q) \rightarrow x$ **Int**$($F $p \wedge$ F $q)$

Această proprietate spune că dacă un agent intenţionează p şi intenţionează q, atunci e normal că el intenţionează realizarea lor într-o ordine temporală nedeterminată: p înainte de q, q înainte de p, sau ambele simultan. Acest lucru este valabil deoarece funcţia **I** asignează fiecărui agent, la un moment dat o singură mulţime de drumuri. Astfel, dacă atât p cât şi q (care sunt formule referitoare la drumuri) vor apărea pe toate drumurile selectate, atunci ele vor apărea într-o anumită ordine temporală pe fiecare din drumurile selectate. Formula (F $p \wedge$ F $q)$ este adevărată la un moment dat pe un anumit drum exact atunci când p şi q sunt adevărate la anumite momente viitoare (posibil distincte) pe drumul ales.

I3 **Persistenţa nu cauzează succesul:**
$\mathrm{EG}((x\ \mathbf{Int}\ p) \wedge \neg p)$ poate fi satisfăcută
(formula EG*p* este validă dacă *p* va fi tot timpul adevărată pe un anumit drum)
Această proprietate este intuitivă: doar pentru că un agent continuă (persistă) cu o anumită intenţie, nu înseamnă că va reuşi. De fapt, din punct de vedere tehnic există două aspecte: pe de o parte, agentul trebuie să ştie cum să-şi realizeze intenţiile, pe de altă parte trebuie să ştie cum trebuie să acţioneze pentru a-şi realiza intenţiile. Cerinţa de a stabili condiţiile în urma cărora agentul îşi va realiza intenţiile (va reuşi) este unul din motivele aparţiei conceptului de "*know-how*".

8.4.4 Angajamente

După cum s-a arătat anterior, scopurile şi intenţiile sunt aproape similare în structura lor semantică. Una din proprietăţile care separă, totuşi, scopurile de intenţii este *angajamentul*.

Un agent este în mod obişnuit tratat ca fiind angajat intenţiilor sale. Astfel de angajamente apar în interiorul unui agent individual dat şi se mai numesc *angajamente psihologice*. Angajamentele unui agent controlează dacă acesta va persista cu intenţiile sale, şi dacă da, cât timp. Există o convenţie, şi anume ca angajamentele să fie tratate ca fiind cele care constrâng modul în care intenţiile sunt revizuite şi actualizate. Angajamentele pot fi descrise pe baza unei noi constrângeri, I4.

I4 **Continuă până când reuşeşti:**
Această constrângere impune ca agenţii să renunţe la a-şi revizui intenţiile, atâta timp cât ei sunt capabili să acţioneze corespunzător. Dacă un agent selectează anumite drumuri, atunci la momente viitoare pe aceste drumuri, el va selecta dintre componentele viitoare ale acestor drumuri. Se notează prin $[a]^x$ mulţimea perioadelor de-a lungul cărora acţiunea *a* este executată de către agentul *x*.
$(S \in \mathbf{I}(x,t)$ şi $[S;t,t'] \in [a]^x) \Rightarrow (\forall S' \in \mathbf{I}(x,t') \Rightarrow (\exists S'' \in \mathbf{I}(x,t)$ şi $S' \subseteq S''))$

Oricum, se consideră că tratarea angajamentelor şi actualizarea intenţiilor, joacă un rol important în abordările bazate pe logică şi cele probabilistice.

8.4.5 Know-How

Intenţiile au o evidentă legătură cu acţiunile - agenţii acţionează pentru a-şi satisface intenţiile. Oricum, intenţiile nu le asigură succesul; proprietatea I3 descrisă anterior, arăta că nici persistenţa nu este suficientă pentru succes. O componentă necesară în acest sens este cea numită *know-how*, componentă care va fi formalizată în continuare.

Exemplu

În Figura 8.2, la momentul t_0, x poate executa sau acţiunea a, sau acţiunea b, atâta timp cât amândouă ar putea practic conduce la unul din drumurile preferate. Oricum, dacă celălalt agent execută acţiunea d, atunci indiferent ce acţiune ar alege x, nu va reuşi cu intenţiile sale, deoarece nici unul din drumurile sale preferate nu va fi realizat.

Vom spune că un agent x ştie cum să realizeze p, dacă este capabil să determine p prin intermediul acţiunilor sale, altfel spus să forţeze apariţia lui p.

Opiniile şi cunoştinţele agentului trebuie să fie luate în considerare, din moment ce acestea îi influenţează decizia. Spre exemplu, dacă un agent este capabil să formeze toate combinaţiile posibile ale unui seif, atunci el va fi capabil să deschidă seiful: în mod sigur, combinaţia corectă va fi printre cele pe care le poate forma. Pe de altă parte, pentru ca un agent să ştie cu adevărat cum să deschidă un seif, nu trebuie numai să aibă abilitatea de a şti sa formeze diferite combinaţii, ci să şi ştie care combinaţie să o formeze.

Un arbore de acţiuni este format dintr-o acţiune, numită rădăcină şi o mulţime de subarbori. Ideea este că agentul efectuează la început acţiunea din rădăcină şi, apoi, alege unul din subarborii disponibili pentru a continua. Cu alte cuvinte, un arbore de acţiuni pentru un agent este proiecţie a unei părţi din **T** pe acţiunile agentului. Vom nota prin **B** o mulţime de simboluri de acţiuni, iar prin **T** o mulţime de momente de timp. Astfel, un arbore include *câteva* din acţiunile posibile ale unui agent dat, alese pentru a determina o condiţie dată. Fie γ mulţimea arborilor. Atunci, γ va fi definit după cum urmează:

$A1$ $\phi \in \gamma$ (ϕ este arborele vid)

$A2$ $a \in \mathbf{B}$ implică $a \in \gamma$

$A3$ $\{\tau_1,...,\tau_m\} \subseteq \gamma, \tau_1,...,\tau_m$ au diferite rădăcini, şi $a \in \mathbf{B}$ implică $< a : \tau_1,...,\tau_m > \in \gamma$

În continuare, se va extinde limbajul formal cu o construcţie auxiliară. Această extensie este destinată simplificării definiţiilor.

$SIN-1$: $\tau \in \gamma$, $x \in \mathbf{A}$, $p \in L_I$ implică $x[(\tau)]p \in L_I$

$x[(\tau)]p$ înseamnă că agentul x ştie cum să realizeze p relativ cu arborele γ. Pentru simplificare, simbolul agentului se va omite dacă se poate deduce din context.

$SEM-4$: $M \models_t [(\phi)]p$ ddacă $M \models_t \mathrm{K}_t p$

$SEM-5$: $M \models_t [(a)]p$ ddacă $M \models_t \mathrm{K}_t(\mathrm{E} < a > true \wedge \mathrm{A}[a]\mathrm{K}_t p)$

$SEM-6$: $M \models_t [(a : \tau_1,...,\tau_m)]p$ ddacă

$$M \models_t \mathrm{K}_t(\mathrm{E} < a > true \wedge \mathrm{A}[a](\vee_{1 \le i \le m} \tau_i : ([(\tau_i)]p)))$$

Menţionăm că formula E<a>p este validă dacă p este adevărată la sfârşitul executării acţiunii a pe un drum, iar formula A[a]p este validă dacă p este adevărată la sfârşitul executării acţiunii a pe orice drum.

Deci, un agent ştie cum să realizeze p urmând arborele vid şi neefectuând nimic, dacă el ştie că p deja este valid. Ca o consecinţă a cunoştinţelor sale, agentul nu va executa nici o acţiune specifică pentru a realiza p. Cazul netrivial de bază este cel în care agentul ştie cum să realizeze p executând o singură acţiune: aceasta ar fi ultima acţiune pe care o execută agentul pentru a realiza p. În acest caz, agentul trebuie să ştie că va şti p imediat după acţiunea dată.

Este important să fie nevoie de cunoştinţe în stările în care agentul în cele din urmă realizează condiţia dată, deoarece acestea îl ajută în limitarea acţiunilor selectate. Dacă p este validă, dar agentul nu ştie lucrul acesta, atunci el ar putea selecta în continuare acţiuni pentru a realiza p.

În cele din urmă, un agent ştie cum să realizeze p urmând un arbore, dacă el ştie că prima dată alege rădăcina arborelui şi apoi, el ar trebui să ştie cum să realizeze p urmând unul din subarborii săi. Astfel *know-how* presupune cunoaştere (cunoştinţe) pentru selectarea următoarei acţiuni şi convingerea că ar trebui să se ştie ce să se facă în momentul în care acţiunea a fost executată.

$SEM-7:\quad M \models_t x\mathrm{K}_h p$ ddacă $(\exists \tau : M \models_t x[(\tau)]p)$