UNIVERSITA' DEGLI STUDI
DI MILANO
DIPARTIMENTO DI SCIENZE
DELL'INFORMAZIONE

BULL ITALIA
CENTRO STUDI
INFORMATICA
E AUTOMAZIONE

Note di Software

Anno 1990

50

Dicembre

Note di software è una pubblicazione trimestrale, edita a cura del Dipartimento di Scienze dell' Informazione dell' Università di Milano e dal Centro Studi Informatica e Automazione della Bull Italia.

Note di software intende costituire uno strumento per la circolazione di informazioni, idee, metodologie ed esperienze nell'area delle tecnologie e della produzione del software.

La collaborazione a **Note di software** è libera. Chi desiderasse pubblicare un contributo è pregato di prendere contatto con il Direttore responsabile o con il Comitato di Redazione. In particolare si sollecitano contributi nella forma di monografie, da inviare in triplice copia, e che non dovrebbero superare le cinquemila parole.
I lavori ricevuti per la pubblicazione verranno revisionati dai membri del Comitato di Revisione Scientifica, che possono avvalersi della collaborazione di specialisti del settore. I lavori pubblicati riflettono comunque le opinioni dei loro autori.

Note di software viene distribuito ad Aziende, Enti e specialisti del settore che ne facciano richiesta alla Redazione.

Direttore Responsabile: Franco Filippazzi
Bull Italia, Centro Studi Informatica e Automazione, via Vida 11 - 20127 Milano

Comitato di Redazione: Stefania Bandini, Francesco Gardin, Roberto Polillo
Università di Milano, Dipartimento di Scienze dell' Informazione
via Moretto da Brescia, 9 - 20133 Milano, tel. (02) 7575233

Comitato di Revisione Scientifica: Alberto Bertoni, Gianluigi Castelli, Giovanni Degli Antoni, Giorgio De Michelis, Mario Italiani, Gaetano Aurelio Lanzarone, Giancarlo Martella, Marco Maiocchi, Giancarlo Mauri.

Autorizzazione del Tribunale di
Milano n. 87 del 23 febbraio 1977

Supporto tecnico - gestionale e distribuzione: Domenico Asinelli
Bull Italia, via Tazzoli 6
20154 Milano, Tel. 02 - 67792960

UNIVERSITA' DEGLI STUDI DI MILANO
DIPARTIMENTO DI SCIENZE DELL'INFORMAZIONE

Bull Italia
CENTRO STUDI INFORMATICA E AUTOMAZIONE

Note di Software

Anno 1990
50
Dicembre

QUESTO NUMERO

Apre questo numero della rivista un articolo di Vieri Sameck-Lodovici, che descrive uno studio sulla integrazione linguistica, visiva e tattile applicata ad un sistema multimediale di immagini fotografiche e filmati.

Nell'articolo successivo Carlo Ferigato e Ivan Maffioli presentano un progetto implementativo di una tecnica di rappresentazione della conoscenza basata su un originale spunto teorico.

Successivamente Massimo Paltrinieri illustra un programma Prolog che permette di risolvere i problemi di soddisfacimento di vincoli mediante algoritmi di filtro con un approccio modulare.

Il quarto lavoro illustra SEDAL, un sistema esperto sviluppato per la diagnostica dei guasti di acceleratori lineari per uso clinico. Autori sono S. Bandini, G. Lavelli, S. Scamuzzo e G. Scielzo.

Infine G. Crosta e A. Paccanaro illustrano l'applicazione di una specifica tecnica di filtraggio alla misura dei campi magnetici cerebrali.

LA REDAZIONE

DEISSI E NLP: DISAMBIGUAZIONE RECIPROCA FRA AZIONI DI PUNTAMENTO GESTUALE ED ESPRESSIONI LINGUISTICHE

Vieri Samek-Lodovici
IRST, Istituto per la Ricerca Scientifica e Tecnologica
Povo (Trento)

RIASSUNTO

In questo lavoro viene descritto il trattamento dei dimostrativi deittici all'interno del sistema multimediale per l'esplorazione dei beni culturali ALFRESCO. Utente e sistema condividono un contesto visivo costituito da immagini fotografiche e filmati di affreschi del Trecento. Il canale di interazione linguistica (modulo di Natural Language Processing) è stato integrato con il canale di interazione visiva e tattile (selezione di figure ed immagini tramite tocco manuale su touchscreen). In particolare viene descritto il processo di disambiguazione incrociata fra espressioni linguistiche contenenti dimostrativi e selezione manuale nel contesto visivo condiviso.

ABSTRACT

In this work the treatment of deietic objects into the multimedial system ALFRESCO for the exploration of cultural goods is described. User and system share a visual context constituted of photos and films about frescos. The channel of linguistic interaction (Natural language Processing module) has been integrated with the channel of visual and tactil interation (touchscreen). The process of crossing disambiguation between linguistic expression and manual selection in the shared visual context is here described.

1. INTRODUZIONE

È sempre più evidente la necessità di *integrare* le interfacce in linguaggio naturale con moduli per la manipolazione diretta delle entità del dominio di interazione. Oggi viene generalmente proposto di offrire una rappresentazione grafica di tali entità, cosicchè l'utente possa direttamente farvi riferimento tramite mouseclicking oppure tocco su touchscreen o ancora DataGlove™.
Molti autori promuovono questo approccio multimediale. Appelt [1] fa notare come puntare a delle rappresentazioni grafiche delle entità in un campo deittico sia più semplice che descriverle. Diversi autori ([25], [19], [23], [10]) sostengono, in particolare, che il puntamento deittico può facilitare il reperimento e la costruzione di insiemi d'oggetti di cui sarebbe difficile la descrizione a priori, quando ancora non se ne conosce né il nome né la composizione. Inoltre vedere la rappresentazione grafica delle entità trattate consente all'utente di farsi una idea più precisa del dominio di interazione ([6]).
Tutti questi argomenti mettono in luce l'insufficienza di un'interazione ristretta allo scambio di espressioni linguistiche. Sorge allora spontanea la domanda: se nell'interazione uomo-macchina, il linguaggio naturale si rivela tanto insufficiente, vale la pena di *integrarlo* con altri moduli, come quello per il puntamento grafico, o non sarebbe meglio *sostituirlo* completamente, abbandonando l'idea di basare sul linguaggio naturale l'interazione con la macchina?

Alcuni studi e considerazioni indicano che la risposta è a favore dell'integrazione. Per esempio, Walker e Whittaker [23] hanno osservato le modalità di intera-

zione di un gruppo di manager con un DataBase dove era possibile passare a propria scelta da un'interrogazione via menù ad una in linguaggio naturale. A dispetto di un tasso sistematico di errori di comprensione da parte del sistema, una parte consistente del gruppo preferiva le interrogazioni in linguaggio naturale a quelle via menù dimostrando come il linguaggio naturale venisse percepito come il mezzo di interazione migliore. Cohen e Sullivan [6] rilevano inoltre come il linguaggio naturale sia più sintetico nell'esprimere informazioni sulla quantificazione e come per mezzo dei pronomi sia più semplice correlarsi al contesto di interazione.

Se la strada della integrazione è quella da percorrere, è naturale partire dai dispositivi linguistici che nel linguaggio realizzano un legame col contesto di interazione. In particolare la deissi spaziale risulta essere il campo più direttamente affrontabile ed affrontato, forse anche perchè, come sostiene Levinson [13], le espressioni linguistiche deittiche sono «il modo più ovvio in cui la relazione fra contesto e linguaggio si riflette nelle strutture del linguaggio stesso».
In questo rapporto viene descritto il trattamento del dimostrativo deittico italiano ''questo'' all'interno del sistema di NLP ALFRESCO[1]. Il sistema integra un modulo di analisi linguistica con un modulo di puntamento manuale su touchscreen alle entità del dominio di interazione. In particolare il trattamento del puntamento è coordinato a quello delle espressioni linguistiche e consente di usare il primo per disambiguare le seconde e viceversa, anche simultaneamente.

2. SISTEMI ESISTENTI: CAPACITÀ E LIMITI

La prima generazione di sistemi di interazione integranti linguaggio naturale e moduli di puntamento è caratterizzata dalla mancanza di ambiguità in fase di selezione: fra il punto indicato dall'azione di puntamento e l'entità ad esso associata vige una relazione predeterminata. La selezione diretta di entità del dominio è dunque indipendente dal contesto linguistico. Le espressioni linguistiche possono utilizzare le entità fornite in output ma non influenzano il precesso di selezione delle stesse o lo influenzano debolmente. Per esempio in NLG ([4]) l'utente può disegnare oggetti geometrici con comandi in linguaggio naturale facendo riferimento al contesto deittico all'interno delle espressioni linguistiche, come in ''Put a point A1 **here**, ma la determinazione della regione da associare a ''**here**'' è indipendente dalla frase enunciata. Altri sistemi[2] simili sono SCHOLAR ([5]), il sistema di Woods [26], SDMS ([3]), NLmenù ([21]) e Language Craft System, ([10]).
Passando a sistemi più recenti, il sistema di NLP ''Chat'' ([6]) estende il ruolo del puntamento deittico alla determinazione del contesto e consente di passare argomenti a comandi specifici sia per mezzo di espressioni in linguaggio naturale sia tramite puntamento alle loro rappresentazioni iconiche. Inoltre utilizza il contesto linguistico per disambiguare il puntamento, tuttavia non tratta la disambiguazione reciproca incrociata. CUBRICON ([14]) sembra fare eccezione, utilizzando il puntamento deittico per disambiguare espressioni in linguaggio naturale e viceversa. Infine TACTILUS ([2], [12]), il cui dominio di interazione è costituito da moduli per la dichiarazione delle tasse, offre all'utente diversi gradi di granularità[3] nella selezione delle entità e consente di avere le regioni direttamente selezionabili l'una innestata nell'altra. Il puntamento ad una regione può fare riferimento sia alla regione spaziale in quanto tale (per introdurre un valore) sia al concetto relativo (per esempio la fonte di reddito). TACTILUS tratta perciò una doppia ambiguità sia grafica (limitata all'inclusione fra diverse regioni) sia concettuale (dualità concetto/regione spaziale).

3. TRATTAMENTO INTEGRATO DEI DIMOSTRATIVI DEITTICI E DELLE AZIONI DI PUNTAMENTO

All'IRST di Trento stiamo sviluppando il prototipo di un sistema per l'accesso in linguaggio naturale ad informazioni sui beni culturali. Il modulo di puntamento è stato integrato con un parser ([20]), un modulo di knowledge representation ([8]) ed altre componenti non direttamente attinenti al lavoro qui presentato.
Il dominio di interazione è costituito da filmati ed immagini relative ad affreschi del Trecento italiano contenuti su un videodisco. Per molti affreschi sono disponibili anche ingrandimenti di alcuni dettagli, solitamente relativi ai personaggi principali dell'opera.

Filmati ed immagini possono essere mostrati su un Touchscreen. Nella base di conoscenza è rappresentato il contenuto degli affreschi, ristretto alle scene principali (p.es. ''annunciazione''), ai personaggi animati (santi, angeli ed animali) e le informazioni generali sul singolo affresco, quali l'autore, il luogo e l'età.

Queste scelte comportano già due importanti innovazioni rispetto ai sistemi esistenti:

1) Come altri sistemi simili anche ALFRESCO offre una zona ''deputata'' alla deissi mutuamente condivisa fra utente e sistema. Tuttavia nel nostro sistema il campo deittico è costituito dalle immagini fotografiche degli affreschi e queste non presentano la natura intrinsecamente bidimensionale delle immagini normalmente proposte da altri sistemi (mappe, moduli delle tasse o dell'università, disegni a icone attive). Inoltre gli affreschi non presentano una partizione rigida dello spazio immediatamente discernibile dall'utente e perciò le possibili sovrapposizioni fra le regioni selezionabili vanno oltre ai rapporti di adiacenza o inclusione diretta.

2) Come anche altri autori fanno notare la sostituzione della selezione via mouse con quella tramite tocco manuale su touchscreen aumenta fortemente la naturalezza dell'atto deittico ([18], [9]). Né le immagini fotografiche proiettate da videodisco su touchscreen sembrano soffrire dei problemi di cattiva risoluzione accennati da D. Schmauks [18] per le immagini proiettate da computer a monitor.

Tornando alla descrizione del sistema, per ogni immagine fotografica vengono determinate, tramite un apposito editor grafico, un insieme di regioni da associare ai personaggi ed alle scene in essa rappresentati, così che dal tocco sul touchscreen si possa reperire l'entità associata. Il processo di reperimento parte dalle coordinate del tocco sul touchscreen. Le coordinate vengono confrontate con ogni regione associata all'immagine corrente per scartare tutte quelle che non le contengono. Per ogni regione rimasta si recupera nella base di conoscenza l'entità (santo, scena, animale, etc...) da essa circoscritta e ad essa associata in fase di editing. In tal modo dal tocco si perviene ad un insieme di entità cui l'utente può aver inteso fare riferimento.
Durante l'interazione in linguaggio naturale, l'utente può fare riferimento direttamente al contesto deittico facendo seguire alla parola ''questo'', all'interno dell'input linguistico, un tocco sull'entità desiderata, completando poi la frase.
Per esempio avendo per contesto deittico l'affresco di Giotto ''Fuga in Egitto'' l'utente può chiedere: ''Chi è questa △ donna?'' toccando nel punto ''b'' indicato in figura, ed il sistema risponde ''la Madonna''. Il segno ''△'' dopo la parola ''questa'' rappresenta simbolicamente il tocco dell'utente, ma non viene digitato né appare nell'input linguistico reale. Serve soltanto a distinguere nella trascrizione il caso precedente dalla richiesta ''Chi è questa donna?'' priva di puntamento diretto nel contesto deittico.
Le regioni dei vari personaggi possono sovrapporsi completamente o parzialmente: per esempio tornando alla figura, la Madonna ha in braccio Gesù bambino, perciò la regione più grande, associata alla Madonna, include una regione più piccola associata a Gesù. Un puntamento a Gesù Bambino in questi affreschi è ambiguo e seleziona entrambi i personaggi (selezione ''a'' in figura).

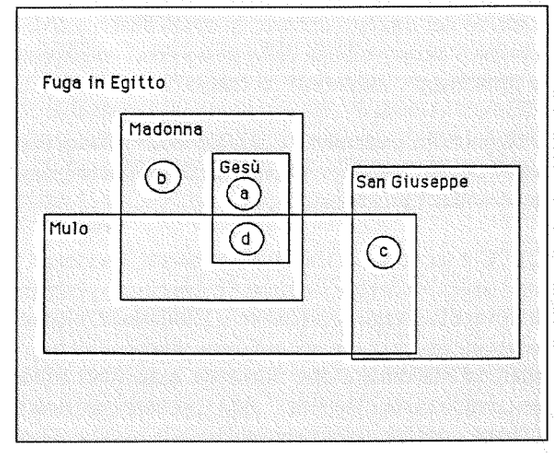

Fig. 1 Regioni associate alle diverse entità nell'affresco ''Fuga in Egitto'' di Giotto.

(1) Il sistema interattivo ALFRESCO (''Automatic Language-Fresco'') è stato sviluppato all'IRST dal Gruppo ''Linguaggio Naturale'' diretto da O. Stock. Il sistema è implementato in Common Lisp e gira su Xerox 1186 e Sun 4. Il sistema utilizza un lettore di videodisco VP405 e touchscreen VP140 Philips. I videodischi ''Storia dell'arte italiana'' sono stati forniti dalla RCS Sansoni Editore - Sidac.

(2) Per una breve descrizione dei sistemi citati in questo paragrafo si veda anche [22].
(3) L'utente può scegliere fra quattro modalità di puntamento. Tuttavia ciò aggiunge un grado di indirettezza nella fase del puntamento che s'aggiunge all'indirettezza di un puntamento via mouse e non gestuale.

L'associazione di regioni e personaggi è stata ripetuta per ogni affresco e per i relativi ingrandimenti. Pertanto il nostro sistema consente di cambiare il contesto deittico mano a mano che il dialogo procede. Inoltre il modulo di rappresentazione della conoscenza mantiene l'identità dei personaggi che possono avere diverse rappresentazioni in affreschi distinti o copie di rappresentazioni negli ingrandimenti di uno stesso affresco. Il sistema consente così di fare richieste quali: ''mostrami un affresco che contiene questo △ santo'' toccando la figura di San Francesco sull'affresco corrente per ottenere un nuovo affresco sempre raffigurante San Francesco.
Facendo riferimento alla figura analizziamo come il sistema permetta disambiguamente fra input linguistico e puntamento deittico. Iniziamo con l'utilizzo dell'input linguistico per disambiguare l'input deittico. Un esempio è la richiesta:

(1) ''Potrei vedere un ingrandimento di questo △ bambino?'' toccando Gesù bambino in braccio alla Madonna (selezione ''d'' in figura). Il tocco seleziona Gesù, la Madonna ed il mulo. Ma la descrizione ''bambino'' in input linguistico permette di scegliere fra le tre entità quella desiderata. La disambiguazione è possibile grazie alla tassonomia di concetti all'interno della base di conoscenza, relativa ai personaggi cui sono associate le immagini negli affreschi: è infatti nella base di conoscenza che si può verificare la compatibilità fra il concetto di ''bambino'', usato nell'espressione linguistica, ed il personaggio ''Gesù bambino'', rappresentato all'interno del contesto deittico. In modo analogo si determina l'incompatibilità dello stesso concetto rispetto ai personaggi ''Madonna'' e ''mulo''.

Il rapporto fra puntamento e descrizione linguistica può anche essere l'inverso, e il primo servire a disambiguare la seconda. La richiesta:

(2) ''C'è un dettaglio che contiene questa △ persona?'' toccando la Madonna (selezione ''b'' in figura) permette di trovare il dettaglio contenente la Madonna. L'espressione linguistica di per sè è ambigua, poichè Gesù bambino, la Madonna e San Giuseppe sono tutte entità descrivibili come ''persone''. Ma il puntamento non è ambiguo, avendo selezionato soltanto la regione associata alla Madonna, risolvendo così l'ambiguità dell'input linguistico.

Infine i due vincoli possono funzionare anche simultaneamente: esaminiamo la richiesta:

(3) ''Mostrami un affresco contenente questa △ persona'' integrata da un puntamento del punto ''c'' in figura. Di per sè ''persona'' è ambiguo, essendo compatibile con tre caratteri dell'affresco: Madonna, Gesù e San Giuseppe. Il tocco a sua volta è ambiguo, selezionando San Giuseppe ed il mulo. L'intersezione dei due vincoli è invece risolutiva: partendo dalle entità selezionate deitticamente, calcoliamo all'interno della base di conoscenza quali di esse possono venire descritte come ''persone'', rimanendo col solo San Giuseppe. Il sistema lascia in ogni caso all'utente la libertà di non utilizzare il puntamento deittico associato ai dimostrativi. Se l'utente non tocca lo schermo il sistema fa semplicemente riferimento al contenuto dell'affresco corrente.

Si attivano gli stessi meccanismi di disambiguazione reciproca che invece di operare sulle sole entità selezionate dal tocco dell'utente prendono in esame tutte le entità rappresentate. Per esempio nel caso (1) sopra descritto l'utente se lo desidera può evitare il puntamento perchè l'affresco contiene un unico bambino. Invece nel caso (2) e nel caso (3) non è possibile determinare il referente a partire dalla sola espressione linguistica. Il sistema realizza così un primo passo verso una maggiore libertà per l'utente nell'utilizzazione dei dimostrativi deittici.

4. SVILUPPO DEL PROGETTO E RICERCHE APERTE

Stiamo lavorando a due estensioni del sistema:

1) affiancare al trattamento di ''questo'' il trattamento di ''quello''. In particolare gli affreschi rappresentano delle scene realistiche e così consentono all'utente di pensare l'immagine dotata di una certa profondità. Viene naturale usare il dimostrativo ''quello'' per chiedere informazioni o ingrandimenti su entità rappresentate sullo sfondo o parzialmente coperte (da entità rappresentate sullo sfondo o parzialmente coperte) da entità in primo piano. Questo uso del dimostrativo ''quello'' non sembra essere stato considerato nei sistemi precedentemente descritti, probabilmente perchè inibiti dalla natura non solo fisicamente ma anche concettualmente bidimensionale dello spazio deittico prescelto.

2) sfruttare la preminenza che l'autore ha voluto dare ad alcune figure rispetto ad altre, tramite una rappresentazione più sofisticata del contenuto degli affreschi. Il grado di preminenza è senz'altro una valutazione soggettiva ma per molti affreschi rivela un suo valore euristico: in caso di ambiguità fra una figura preminente ed un'altra verrebbe selezionata la prima, nell'ipotesi che l'utente ne percepisca la preminenza e senta di dover fornire l'informazione necessaria alla disambiguazione soltanto per la selezione della seconda entità, in accordo con la massima di rilevanza di Grice.

È invece già realizzato in buona parte l'estensione del trattamento dei dimostrativi ad altre fonti di referenti. In particolare è possibile riferirsi ad entità menzionate nel corso del dialogo grazie ad un meccanismo di gestione dell'attenzione ([17]).
Altri aspetti di puntamento deittico da noi non trattati ma in parte già presenti in altri sistemi riguardano la selezione diretta -via touch- anche per i sintagmi nominali definiti privi di dimostrativi, il trattamento delle relazioni spaziali fra i vari referenti ([16], [27]), il trattamento di immagini dinamiche [27], la possibilità di puntare con azioni differenziate (come il tocco continuo e strisciato oppure circondando le entità desiderate ([11], [22]), l'integrazione con input vocale ([9]). Rispetto invece alle funzionalità dei dimostrativi non trattiamo la referenza clausale ([24], [7]) e l'uso enfatico ([15]).

BIBLIOGRAFIA

[1] D.E. Appelt, PLANNING ENGLISH SENTENCES, Cambridge University Press, 1985.

[2] J. Allgayer, C. Reddig, PROCESSING DESCRIPTIONS CONTAINING WORDS AND GESTURES - A SYSTEM ARCHITECTURE, in Rollinger (ed) Proc. of GWAI/ÖGAI, Berlino, 1986.

[3] R.A. Bolt, PUT THAT THERE: VOICE AND GESTURE AT THE GRAPHICS INTERFACE, Computer Graphics, 14, pp 262-270, 1980.

[4] D.C. Brown, S.C. Kwasny, B. Chandrasekaran, N.K. Sondheimer, AN EXPERIMENTAL GRAPHICS SYSTEM WITH NATURAL LANGUAGE INPUT, Computer and Graphics, A, pp. 13-22, 1979.

[5] J.R. Carbonell, MIXED INITIATIVE MAN-COMPUTER DIALOGUES, BBN Rep. N. 1971. BBN, Cambridge MA USA, 1970.

[6] P.R. Cohen, J.W. Sullivan, SYNERGETIC USE OF DIRECT MANIPULATION AND NATURAL LANGUAGE, Proc. of CHI-89 ''Human Factors in Computer Systems''. Austin, Texas USA, May 1989.

[7] B. DiEugenio, CLAUSAL REFERENCE IN ITALIAN, Penn. Linguistics Colloquium, U. of Pennsylvania USA, 1989.

[8] E. Franconi, B. Magnini, O. Stock, KRAPFEN - PROGRESS REPORT, IRST, Trento, Italy, October 1988.

[9] A.G. Hauptmann, SPEECH AND GESTURES FOR GRAPHIC IMAGE MANIPULATION, Proc. of CHI-89 ''Human Factors in Computer Systems'', Austin, Texas USA, May 1989.

[10] P.J. Hayes, STEPS TOWARD INTEGRATING NATURAL LANGUAGE AND GRAPHICAL INTERACTION FOR KNOWLEDGE BASED SYSTEMS, Proc. of ECAI, Brighton UK, 1986.

[11] J.C. Jackson, R.J. Roske-Hofstrand, CIRCLING: A METHOD OF MOUSE-BASED SELECTION WITHOUT BUTTON PRESSES, Proc. of CHI-89 ''Human Factors in Computer Systems'', Austin, Texas USA, May 1989.

[12] A. Kobsa, J. Allagyer, C. Reddig, N. Reithinger, D. Schmauks, K. Harbusch, W. Wahlster, COMBINING DEICTIC GESTURES AND NATURAL LANGUAGE FOR REFERENT IDENTIFICATION, Proc. of the 11th Conf. on Computational Linguistics, Bonn RFT, 1986.

[13] S.C. Levinson, PRAGMATICS, Cambridge University Press, 1983.

[14] J.G. Neal, S.C. Shapiro, INTELLIGENT MULTIMEDIA INTERFACE TECHNOLOGY, Proc. on the Workshop on Architectures for Intelligent Interfaces: Elements and Prototypes, Monterey CA USA, 1988.

[15] L. Renzi, GRANDE GRAMMATICA ITALIANA DI CONSULTAZIONE, Vol. 1, il Mulino, Bologna, 1988.

[16] G. Retz-Schmidt, VARIOUS VIEWS ON SPATIAL PREPOSITIONS, VITRA Project, Bericht N 33, Universität des Saarlandes, RFT, 1988.

[17] V. Samek-Lodovici, C. Strapparava, INTEGRATING DEICTIC AND LINGUISTIC REFERENCES: THE TOPIC MODULE OF THE ALFRESCO SYSTEM, IRST-Tech. Rep. 9002-10, Trento, Italy, 1990.

[18] D. Schmauks, NATURAL AND SIMULATED POINTING: AN INTERDISCIPLINARY SURVEY, XTRA Project, Bericht N. 16, Universität des Saarlandes, RFT, 1987.

[19] P. Stenton, D. Proudian, S. Whittaker, D. Adger, SUPPORTING SET MANIPULATION DIALOGUES FOR INFORMATION RETRIEVAL, Tech. Rep. HPL-BRC-TM-89-020, HP labs, Bristol, UK, 1989.

[20] O. Stock, PARSING WITH FLEXIBILITY, DINAMIC STRATEGIES AND IDIOMS IN MIND, Computational Linguistics 15(1):1-18, 1989.

[21] C. Thompson, BUILDING MENU-BASED NATURAL LANGUAGE INTERFACES, Texas Engineering Journal, 3, pp 140-150, 1986.

[22] W. Wahlster, USER AND DISCOURSE MODELS FOR MULTIMODAL COMMUNICATION, XTRA Project, Bericht N. 37, Universität des Saarlandes, RFT, 1988.

[23] M. Walker, S. Whittaker, WHEN NATURAL LANGUAGE IS BETTER THAN MENUS: A FIELD STUDY, Tech. Rep. HPL-BRC-TM-89-020, HP labs, Bristol, UK, 1989.

[24] B.L. Webber, DISCOURSE DEIXIS: REFERENCE TO DISCOURSE SEGMENTS, Proc. of the ACL annual meeting, U. of Buffalo, New York USA, 1988.

[25] S. Whittaker, P. Stenton, USER STUDIES AND THE DESIGN OF NATURAL LANGUAGE SYSTEMS, Proc. of the 4th Conference of the European Chapter of the ACL, Manchester, UK, 1989.

[26] W.A. Woods et al., RESEARCH IN NATURAL LANGUAGE UNDERSTANDING, Annual Report, Tech. Rep. 4274. BBN, Cambridge MA USA, 1979.

[27] G. Zimmermann, C.K. Sung, G. Bosch, J.R.J. Schirra, FROM IMAGE SEQUENCES TO NATURAL LANGUAGE: DESCRIPTION OF MOVING OBJECTS, VITRA Project, Bericht N 17, Universität Saarlandes, RFT, 1987.

UN PROGETTO DI IMPLEMENTAZIONE DEL MULTIVERSUM DELLA CONOSCENZA

Carlo Ferigato, Ivan Maffioli
Dipartimento di Scienze dell'Informazione
Università degli Studi di Milano

RIASSUNTO

Il "Multiversum della Conoscenza" è un progetto di realizzazione di una teoria e di una tecnica di rappresentazione della conoscenza in corso di sviluppo presso il Dipartimento di Scienze dell'Informazione dell'Università degli Studi di Milano. Il lavoro è fondato sia su considerazioni di carattere epistemologico che su argomenti inerenti la pragmatica della comunicazione umana elaborati da Giorgio De Michelis nel corso di studi e di esperienze sulla comunicazione in gruppi sociali ristretti. La parte teorica consiste nel "rovesciare" l'atteggiamento dominante in questa disciplina, considerando centrale non più la possibilità di rappresentare la conoscenza facente capo al singolo individuo, tipica di una concezione "forte" dell'intelligenza artificiale, ma quella di rappresentare la conoscenza condivisa dai componenti di un gruppo sociale. Dal punto di vista della tecnica di rappresentazione la base di conoscenze progettata non considera più 'vero' o 'falso' ciò che la sua struttura ed il suo contenuto rappresentano di un'approssimazione del mondo fatta da chi la costruisce, ma piuttosto ciò che lo stesso gruppo sociale considera correttamente esprimibile, ottenendo, tra l'altro, la possibilità di un continuo aggiornamento senza compromettere la consistenza.

ABSTRACT

"Multiversum della Conoscenza" is the name of a Knowledge Representation project that the Computer Science Department of the Milano University is developing. The main idea consists in changing the traditional point of view, founded on the representation of individual knowledge, for a shared conception; shared among the members of a social group. The foundations of this work are in the fields of epistemology (from Searle's and Winograd's work against "strong" conceptions of Artificial Intelligence), in philosophy (from Austin's and the last Wittgenstein's work about language) and in the experience of G. De Michelis about the pragmatic of communication in small social systems.

1. INTRODUZIONE

A dispetto della grande evoluzione dell'insieme di procedimenti di carattere euristico noti come "intelligenza artificiale", il problema di cosa significhi, nell'ambito di un sistema formale, "rappresentare conoscenza" resta tuttora aperto. Il nostro lavoro (iniziato come tesi di laurea [2, 8, 13]), vuole inserirsi in modo costruttivo in questo contesto ed affrontare la questione prendendo spunto dal pensiero e dall'opera di due grandi filosofi contemporanei: John Langshaw Austin e Ludwig Wittgenstein.
Per entrambi è essenziale il considerare prioritario l'aspetto pragmatico della comunicazione umana, nel senso che su questo è possibile fondare gli aspetti semantici, e non viceversa. Per Wittgenstein il linguaggio è costituito da un numero imprecisato di 'giuochi linguistici'[1], e per questo:
"L'impiego del linguaggio, da cui dipende il significato, consiste in un insieme di attività multiformi e intrecciantisi sia reciprocamente, sia con gli altri tipi di

(1) Per la definizione di 'giuoco linguistico' v. [21], §§7,23,71.

attività sociale; e fra di esse non è legittimo cercare un''essenza' comune, né privilegiarne una come primaria. La comunicazione d'informazioni ne è una soltanto, neppure la più importante.[2]'',

All'interno dei giuochi linguistici, inoltre, i concetti di 'verità' e 'falsità' di un'affermazione assumono un significato diverso da quello usuale: non ha più senso chiedersi ''che cosa deve darsi perchè questo enunciato sia vero?'' ma piuttosto le domande corrette divengono ''In quali circostanze questa espressione può essere asserita (o negata) appropriatamente?'' e ''Quale ruolo svolge nella nostra vita e quale utilità vi ha la pratica di asserire (o negare) l'espressione in queste condizioni?'' ed è da notare che le domande non sono più applicabili solamente ad enunciati di tipo assertivo ([12] ed. it. p. 62). Per Austin è preponderante l'esigenza di distinguere e definire diversi tipi e livelli ''d'uso del linguaggio'' al fine di costruire uno schema generale di interpretazione dei fenomeni linguistici (questo anche per contrastare la diffusione di usi generici ed imprecisi di tale concetto seguita alla scelta di metodo operata da Wittgenstein di non definire a priori limiti alla eterogeneità e molteplicità dei giuochi linguistici). Questo lo porta a definire tre classi di 'atti linguistici' (locutori, illocutori e perlocutori), a proporre una tassonomia per gli illocutori ed a parlare di condizioni di 'felicità' o di 'infelicità' per atti di tutte e tre le classi [1].
Dunque, indipendentemente dalle scelte di metodo, per entrambi è essenziale considerare primario il punto di vista dell'impiego del linguaggio e questo implica anche una definizione diversa dei concetti di 'vero' e 'falso' associabili ad un'espressione linguistica.

2. IL MULTIVERSUM DELLA CONOSCENZA

Il ''Multiversum della Conoscenza'' si basa sulle seguenti assunzioni: 1) la dimensione pragmatica della comunicazione umana è (logicamente) primaria; 2) i gruppi sociali determinano e sono determinati dai giuochi linguistici che in questi sono giocati, le regole dei giuochi individuano le mosse comunicative tramite le quali i componenti del sistema interagiscono; (il linguaggio viene considerato unificante per i sistemi sociali anche, a partire da un punto di vista completamente diverso, da Humberto Maturana Francisco Varela che arrivano a considerare il linguaggio strumento di una ''trofallassi sociale'' [14]).
3) È possibile distinguere, nell'ambito della molteplicità dei giuochi linguistici, tre classi di questi: quella costituita dai giuochi linguistici pragmatici, quella formata dai giuochi linguistici confidenziali e quella comprendente i giuochi linguistici semantici[3]. La prima classe contiene quei giuochi che danno senso all'inter-azione umana, quelli nei quali gli uomini si impegnano reciprocamente ad azioni future e si differenziano per i ruoli che occupano in un sistema sociale ''produttivo''. La seconda contiene quei giuochi nei quali gli uomini si danno fiducia reciproca, fanno riferimento alle aspettative di soddisfacimento degli impegni presi; questi sono caratteristici di sistemi sociali, nei quali gli uomini ''convivono''. L'ultima, fondamentale per il nostro lavoro, contiene quei giuochi che la necessità e la prassi del comunicare hanno destinato ad essere il sostrato comune che consente agli uomini di riferirsi alle cose. Esempi di sistemi sociali nei quali i giuochi linguistici pragmatici sono predeterminanti possono essere gli uffici o comunque i gruppi di lavoro, mentre di sistemi sociali associabili principalmente a giuochi linguistici confidenziali possono essere esempi le famiglie. Ma a tutti i sistemi sociali debbono poter essere associati dei giuochi linguistici semantici dato che il comune riferisi alle cose è comunque indispensabile.

Sulla base delle tre assunzioni fatte il nostro progetto di costruzione di una tecnica di rappresentazione consiste nel dare una formalizzazione del giuoco linguistico semantico. Questo, nella nostra ipotesi di lavoro, è caratterizzato da una ''base di conoscenze'' condivisa dagli attori del giuoco: la comprensione di una frase avviene solo tramite riferimento ad un insieme di conoscenze condivise dal gruppo.
Questa base di conoscenze che, istante per istante, identifica tutti i significati possibili delle espressioni effettivamente enunciate e che da queste è incessantemente aggiornata (ogni partecipante al giuoco è membro anche

(2) [16], p. 14.
(3) Giorgio De Michelis, a seguito di esperienze e studi, svolti da lui e dal gruppo di lavoro al quale collabora sulla pragmatica della comunicazione nell'ambito di quei particolari sistemi autopoietici del terz'ordine che sono gli uffici (v. ad es. [5]), ha ampiamente motivato la possibilità della costruzione di tale partizione oltre alla possibilità di associare un giuoco linguistico ad un sistema sociale. Cfr. [6, 7].

di altri gruppi sociali e da questi porta nuove espressioni e nuovi significati) è l'obiettivo del nostro tentativo di rappresentazione della conoscenza. Riteniamo che la base di conoscenze relativa al giuoco linguistico semantico sia rappresentabile come composizione di cinque basi di conoscenze, relative ad 'eventi linguistici', compito delle quali è delimitare un particolare 'universo del discorso': l'**Universo Fisico**, che contiene le rappresentazioni di movimenti e relazioni spaziali ed in sostanza fissa il ''quadro'' spaziale entro cui è possibile muoversi e situare oggetti; l'**Universo Operativo**, che contiene le rappresentazioni delle operazioni e degli strumenti necessari per compierle e circoscrive per questo le possibilità di agire per trasformare materiali; l'**Universo Procedurale**, che contiene la rappresentazione della ''pragmatica dell'agire'' costituendo così il luogo nel quale vengono rappresentati tutti gli svolgimenti possibili dei processi di trasformazione di stato degli oggetti e degli attori in questi coinvolti.
Ma anche dei giuochi linguistici pragmatico e confidenziale è possibile ''parlare'' nel giuoco linguistico semantico, quindi deve essere possibile rappresentare anche l'**Universo degli Impegni**, che contiene le rappresentazioni degli 'atti linguistici' e che definisce proprio la pragmatica del comunicare rappresentando le relazioni che legano gli individui appartenenti a gruppi sociali orientati a produrre e l'**Universo Decisionale** (o Soggettivo), che contiene le rappresentazioni degli stati (psicologici) interni che possono essere espressi da chi parla e traccia per questo il confine entro il quale sono contenuti i giudizi, le valutazioni personali e gli stati d'animo. Questi cinque 'universi' definiscono lo 'spazio della rappresentazione'.

Il nostro tentativo consisterà, inizialmente, nel formalizzare i soli universi fisico, operativo e procedurale in una base di conoscenze che abbia il compito di indicare in che senso una mossa, in un particolare giuoco linguistico semantico può essere 'felice' (accettabile). È da osservare che una base di conoscenze costruita secondo questi presupposti può contenere anche elementi contraddittori senza per questo essere inconsistente: in un giuoco linguistico possono darsi espressioni in contrasto egualmente accettate dai partecipanti.
Il ''Multiversum'' risulta perciò essere un progetto di rappresentazione ''costruttiva'' della conoscenza il cui obiettivo è definire un sistema formale per la base di conoscenze associabile al giuoco linguistico semantico.
Tale sistema è costruttivo in due sensi: in primo luogo ogni nuova frase effettivamente pronunciata si proietta negli universi del discorso determinando 'hic et nunc' la sua rappresentazione in (per ora) tre basi di conoscenze, generando così una successione di rappresentazioni di vari aspetti dello stesso oggetto, visto da tre punti di vista diversi; in secondo luogo ogni frase sulla 'verità' della quale i partecipanti al giuoco concordano, determina una modificazione o una riorganizzazione dei 'dizionari' delle basi di conoscenze per tener conto delle eventuali nuove possibilità di espressione in questa contenute.
Nello spazio del problema una cesura ''orizzontale'' separerà il mondo costruito 'hic et nunc' dall'insieme delle classificazioni delle espressioni linguistiche che possono costruirlo (separerà le mosse fatte in questa partita dalle regole che le determinano), questi oggetti verranno chiamati rispettivamente Hic et Nunc e Dizionario, ed una frammentazione ''verticale'' originata dai tre Universi del Discorso che, per ora, prendiamo in considerazione definirà le seguenti partizioni: Universo Fisico, Universo Operativo, Universo Procedurale (figura 3).

3. IL LIVELLO ''DIZIONARIO''

Il ruolo del dizionario all'interno del Multiversum è quello di creare lo spazio delle possibilità d'uso dei termini. Questo spazio di possibilità permette e guida la costruzione di diversi scenari Hic et Nunc secondo le mosse che il gruppo sociale che condivide il dizionario accetta. La struttura del dizionario è conseguenza dell'approccio seguito: utilizzare cioè l'apparato concettuale degli universi del discorso. La ripartizione in universi: ''universo fisico'', ''universo operativo'' e ''universo procedurale'' viene mantenuta nel dizionario operando una separazione tra gli oggetti rappresentati, separazione messa in evidenza anche dai diversi formalismi utilizzati.

3.1 Il dizionario fisico

Il dizionario fisico è la base di conoscenze contenente i termini che si riferiscono a luoghi, oggetti, movimenti e relazioni spaziali tramite i quali è possibile comporre una ''descrizione dello spazio'' (scena), questa consiste in un elenco degli oggetti e dei siti che caratterizzano quello spazio e nelle loro reciproche relazioni.

I luoghi sono nomi di porzioni dello spazio ai quali si fa riferimento allo scopo di indicarne, con una certa precisione, una parte. Possono essere raggruppati in classi ed a loro volta partizionati in sottoluoghi (zone).
Per un luogo ha quindi senso indicare se questo è zona di un altro, se a sua volta è stato diviso in zone, la classe dei siti alla quale eventualmente appartiene. Gli oggetti sono nomi di entità che occupano luoghi. Allo stato attuale del progetto si è ritenuto significativo indicare per ogni oggetto quelle che sono le *componenti strutturali* e le *componenti spaziali*. Con il primo termine si intendono tutte le parti (a loro volta altri oggetti) che concorrono a formare il tutto; con il secondo si intendono quelle parti di spazio determinate dalla natura particolare dell'oggetto, lo sono ad esempio, la superficie esterna, il volume interno delimitato dalla superficie esterna e occupato dal materiale dal quale quell'oggetto è costituito, le eventuali parti di spazio dovute alla forma come incavi, interni e così via. Viene naturale pensare ai luoghi ed agli oggetti in termini di *frames* anche se una trascrizione dei concetti così espressi in un linguaggio logico al prim'ordine, oltre a mantenere la stessa ''potenza espressiva'' [10] risulta più coerente con quanto verrà detto a proposito delle relazioni spaziali e facilita il compito di definire una semantica per il sistema formale proposto [8].

Un frame per un luogo ha la forma seguente:

(nome_del_luogo
  (**isa**: (<una lista di superclassi>))
  (**zones**: (<una lista di sottoluoghi>))
  (**zone_of**: (<nome di luogo>))
)

e per un oggetto:

(nome_di_oggetto
  (**isa**: (<lista di superclassi>))
  (**parts**: (<lista di componenti strutturali>))
  (**part_of**: (<nome di oggetto>))
  (**s_parts**: (<lista di componenti spaziali>))
)

3.1.1 Relazioni spaziali

Tramite le relazioni spaziali, che associano luoghi a oggetti e oggetti e luoghi tra loro, viene fornito un modo per descrivere particolari configurazioni dello spazio. Si è cercato di individuare un piccolo insieme di relazioni che consenta però di esprimere un buon numero di situazioni possibili. La descrizione di una scena contiene comunque quelle ambiguità che sono tipiche delle descrizioni orali che fanno uso di preposizioni locative (in, su, sopra, vicino a...) senza avvalersi di alcuna metrica. L'insieme individuato consta delle seguenti relazioni: contatto, supporto, vicinanza spaziale, allineamento nelle direzioni verticale e orizzontale, contenimento. Contatto tra due oggetti *a, b* è espresso dalla relazione **Contact** mediante [*a* **Contact** *b*] che sussiste quando i due oggetti si toccano in almeno un punto. Supporto tra due oggetti *a, b* è espresso dalla relazione **Support** mediante [*a* **Support** *b*] che sussiste quando *a* impedisce che *b* si muova in balia della forza gravitazionale che si assume agire nello spazio nel quale essi sono immersi (alcuni esempi possono essere: un vaso su di un tavolo oppure un quadro fissato ad un chiodo). Vicinanza spaziale tra due oggetti o luoghi *a, b* espressa dalla relazione **By** mediante [*a* **By** *b*]. Il concetto di vicinanza è qualcosa di non ben definito, di sfumato; alcuni sistemi definiscono come 'vicino' un oggetto che si trova al di sotto di una certa 'distanza soglia' da un altro; qui (visto che non si assume alcuna metrica) 'vicino' ha il significato intuitivo che ognuno può attribuire come contiguità spaziale.
L'oggetto (o il luogo) che si trova vicino ad un altro giace nella (o è parte della) regione di spazio caratterizzata da una sfera ipotetica di raggio imprecisato centrata in quest'ultimo. Allineamento nella direzione verticale e orizzontale tra due oggetti o luoghi *a, b* è espresso dalle relazioni **Over** e **Side** rispettivamente con: [*a* **Over** *b*] ed [*a* **Side** *b*]. Con queste relazioni si è inteso ''catturare'' i concetti di 'sopra', 'a fianco', cioè allineamenti nelle direzioni parallele ('sopra') e perpendicolari ('a fianco') alla forza gravitazionale. Il contenimento di un oggetto *a* in un luogo *b* è espresso dalla relazione **In** con [a **In** b] che sussiste quando l'oggetto *a* occupa in tutto o in parte la regione di spazio identificata da *b*[1].

(1) Bisogna notare come anche nel linguaggio si usa una preposizione che significa contenimento, come ''in'' o i suoi composti come ''nel'', anche quando l'oggetto inteso è contenuto solo in minima parte nel luogo indicato. L'esempio più significativo è l'affermazione che le piante di un appartamento sono contenute nei vasi, quando sono solo le radici ad esserlo. Per un tentativo di spiegazione cfr. [11] ove si trovano numerosi altri esempi.

Nell'universo fisico, dopo la definizione di un opportuno calcolo, sono possibili deduzioni in base alle proprietà delle relazioni che sono state appena definite ed a certe proprietà che si assume valgano nello spazio, queste costituiscono una sorta di ''assiomi per una teoria dello spazio fisico'' e debbono essere parte del dizionario. Alcuni assiomi stabiliscono la riflessività la simmetria e la transitività delle relazioni spaziali, altri stabiliscono implicazioni in base alle forme degli oggetti, di seguito ne sono riportati alcuni.

Si indicherà con OBJ la classe ''origine'' di tutti gli oggetti:
(OBJ
  (**isa**: ())
  (**parts**: ())
  (**part_of**: ())
  (**s_parts**: (SUPERFICIE_ESTERNA,
    VOLUME_INTERNO))
).
Sia inoltre definita la relazione **Isa** che lega oggetti e classi di oggetti.
Allora si possono dare le seguenti:

A1. [x **Contact** x]
A2. [x **Contact** y] → [y **Contact** x]
A2a. [x **Contact** y] → ([x **Isa** OBJ] & [y **Isa** OBJ])
A3. [x **Support** x] —
A4. [x **Support** y] → [x **Contact** y]
A4a. [x **Support** y] → ([x **Isa** OBJ] & [y **Isa** OBJ])
.
.
.
(etc.)

3.1.2 Movimenti

Oltre alle relazioni spaziali, gli eventi linguistici principali dell'Universo fisico sono i movimenti. Una frase contenente un verbo di movimento, quando viene pronunciata, evoca una serie di cambiamenti della scena fisica. Quando ciò avviene la base di conoscenza di quel giuoco linguistico si aggiorna mutando la validità delle relazioni spaziali verificandone di nuove e falsificandone altre in maniera non monotona. Esprimere un movimento è difficile, esso infatti è un continuum ed adottare l'approssimazione consistente nel costruire (come è d'uso in fisica) una funzione posizione dipendente dal tempo, vincola il nostro parlare dello spazio (e del moto) ad una metrica precisa[2], implica l'adozione di un concetto ben fondato di tempo. Come catturare quindi il concetto di azione nell'universo fisico tramite una struttura dati inesorabilmente statica?
In [17] viene suggerito di considerare le seguenti informazioni:

source: posto dal quale una cosa si muove
path: luogo attraverso il quale una azione occorre
target: il luogo verso il quale una cosa si muove
location: il posto nel quale è incentrata una azione

Delle voci citate si sono ritenute indispensabili la prima e la terza con l'aggiunta di **actor**: l'attore del movimento, e dell'elenco delle relazioni spaziali verificate e di quelle falsificate a seguito del movimento. Tali insiemi di predicati variano in modo continuo formando una sequenza di scene simile a quella di una pellicola cinematografica. I ''fotogrammi'' però sono soltanto due: quello relativo al *prima* (l'evento deve ancora accadere) e quello relativo al *dopo* (azione conclusa)[3]. Chiameremo, in accordo con la letteratura in materia, **addlist** il 'dopo' e **deletelist** il 'prima'.

A livello di dizionario però, non si ha conoscenza del contesto in cui ha luogo un movimento, è solo possibile fornire una struttura (un frame) con le informazioni che si possono ricavare a questo livello. Come esempio si consideri il movimento espresso tramite il verbo ENTRARE. L'attore del movimento può essere un oggetto fisico. L'attore arriva da un luogo che qui non può essere meglio precisato e, ad azione conclusa, si troverà all'interno del luogo denominato **target**, si avrà quindi una **addlist** contenente un predicato [. **In** .], ovviamente l'attore non si trova più nel luogo di origine:

(2) A meno di non partecipare a particolari giuochi linguistici, non è possibile dare un concetto preciso di posizione se non facendo riferimento a relazioni tra entità spaziali (vicino a..., sopra il..., a fianco di... etc...).
(3) Questo modo di porre le cose è tipico delle soluzioni al *frame problem* nei sistemi di planning della generazione STRIPS (v. ad esempio [9], [15]).

(ENTRARE
  (**source**: (<un luogo>))
  (**target**: (<un luogo>))
  (**actor**: (<una lista di oggetti>))
  (**addlist**: ([**actor In target**]...))
  (**deletelist**: ([**actor In source**]...))
)

3.2 Il Dizionario Operativo

Il dizionario operativo è la base di conoscenze che contiene i termini che si riferiscono a strumenti e operazioni che questi rendono possibili, effettuate su materiali. Questa ha come scopo principale quello di fornire alla componente asserzionale del Multiversum (l'Hic et Nunc), un inventario di operazioni e strumenti possibili che indichino quali azioni sono possibili, su quali materiali e con quali strumenti. Anche per strumenti, operazioni e materiali, è stata giudicata adeguata una rappresentazione a frames; un frame per uno strumento è così rappresentato:

(nome_di_strumento
  (**isa**: (<lista di superclassi>))
  (**function**: (<lista di operazioni>)
  (**functional_parts**: (<lista di parti funzionali>))
  (**is_part_of**: (<nome di strumento principale>))
)

lo slot **isa** struttura gli strumenti in una tassonomia, gli slot **functional_parts** e **is_part_of** definiscono una gerarchia di parti funzionali, cioè di strumenti e sottostrumenti.

Il frame per una operazione ha la forma seguente:

(nome_di_operazione
  (**object**: (<lista di materiali>))
  (**instrument**: (<lista di strumenti>))
  (**cons**: (<true o false>))
)

Gli slot **object** e **instrument** indicano, rispettivamente, i materiali sui quali si opera e gli strumenti che possono essere utilizzati, lo slot **cons** indica se l'operazione consuma il materiale su cui opera oppure se quest'ultimo sarà ancora disponibile.

3.3 Il Dizionario Procedurale

Il dizionario procedurale ha il compito di esprimere lo spazio delle possibilità delle trasformazioni di stato relative a procedure nei termini di causalità e concorrenza. Esso deve quindi permettere l'espressione di conoscenza procedurale in qualche formalismo che consenta l'espressione di sistemi concorrenti anche a diversi livelli di astrazione. Molti sono i linguaggi sviluppati per tali scopi, la scelta è caduta sulle reti di Petri ed in particolare sulla classe delle reti ad automi sovrapposti [4]. Le reti di Petri sono un formalismo con solide basi matematiche, rendono pertanto più agevole l'implementazione delle specifiche, posseggono una rappresentazione grafica, caratteristica importante per una buona usabilità, distinguono in modo netto ed esplicito le due nozioni correlate di stato e cambiamento di stato (transizione o azione) ed i vincoli di causalità tra questi senza introdurre alcuna dimensione temporale. Permettono inoltre, di esprimere in modo semplice e chiaro le nozioni di *sequenza, scelta* (o conflitto) e *concorrenza.*

In particolare, la sottoclasse delle reti di automi sovrapposti (reti SA) permette di rappresentare in modo chiaro diverse entità agenti in modo autonomo e tutti i vincoli di sincronizzazione fra queste quando interagiscono reciprocamente. Inoltre, per le reti SA, sono ben definite delle nozioni di raffinamento e di equivalenza che consentono la specifica di sistemi concorrenti a vari livelli di dettaglio senza cambiare linguaggio.

Una rete SA è una rete <S,T,F> che soddisfa le seguenti condizioni:
1) ogni transizione è bilanciata, cioè
$\forall t \in T : I \bullet t I = I t \bullet I \geqslant 1$
2) S può essere ripartito in m classi disgiunte $\pi_1 \ldots \pi_m$ in modo tale che ogni transizione abbia un solo posto di ingresso e un solo posto di uscita appartenente a ciascuna classe.

La sottorete generata da una rete SA N, ottenuta prendendo l'insieme dei posti appartenente ad una partizione π_i di N e le transizioni ad essi collegate viene chiamata *macchina a stati*. Una macchina a stati modella il comportamento di un singolo attore (agente), le transizioni sono azioni locali ed i posti sono stati dell'attore. Una rete SA modella un sistema costituito da più componenti interagenti, ciascuna modellata con una

macchina a stati dal comportamento sequenziale e non deterministico. Queste componenti interagiscono tra loro sincronizzandosi su alcune transizioni dette di interazione. Si chiama T-composizione l'operazione che consente di costruire sistemi SA a partire da componenti elementari o da sistemi SA più semplici. La T-composizione avviene per sovrapposizione di transizioni di interazione. (Per la formalizzazione precisa delle operazioni di T-composizione e di S-composizione e altri termini usati nel seguito si rimanda alla letteratura.)

L'uso di sistemi SA consente di specificare agevolmente delle procedure a diversi livelli di dettaglio. È possibile cioè esprimere un sistema o parte di esso con un altro più dettagliato ma equivalente al primo. Per definire l'equivalenza tra due sistemi bisogna identificare degli stati rilevanti (dette marcature osservabili) comuni ad entrambi. I due sistemi saranno equivalenti se durante la loro evoluzione transiteranno per le stesse marcature osservabili (avranno cioè lo stesso passo elementare S-osservabile), anche se tra una marcatura osservabile e l'altra possono evolvere con modalità diverse. Tale equivalenza rispetto alle marcature osservabili viene indicata in letteratura con il termine EF-equivalenza. Una rete SA potrà quindi contenere alcune transizioni che possono essere ulteriormente raffinate, il raffinamento avviene sostituendo la transizione da dettagliare con la rete SA ad essa EF-equivalente catalogata nel dizionario con lo stesso nome, cioè tramite una operazione di S-composizione tra la rete SA originale privata della transizione da raffinare e la rete che rappresenta il nuovo passo elementare S-osservabile.

3.3.1 Etichettatura

Una rete del dizionario precedurale ha i posti e le transizioni etichettate, l'etichetta ha un duplice scopo: permettere la comprensione della procedura che si vuole modellare, ed evidenziare delle informazioni che facilitino, a livello *Hic et Nunc*, le correlazioni tra le informazioni che provengono dai dizionari.
Le etichette che vengono attribuite alle transizioni sono di tre tipi:
a) Nomi di operazioni compiute dell'attore per mezzo di strumenti su dei materiali.
b) Movimenti dell'attore in luoghi determinati.
c) Azioni complesse (procedure) svolte dall'attore.
Una distinzione delle etichette induce una distinzione tra le transizioni alle quali vengono assegnate. Questa differenziazione però è puramente un artificio pratico, infatti non esiste operazione che non sia compiuta effettuando qualche movimento e d'altra parte, non vi è movimento che non sia compiuto per mezzo di qualche strumento. Con l'etichetta si vuole indicare quale è la trasformazione dello stato significativa che viene modellata nella rete con quella transizione, ad esempio se è la trasformazione compiuta dallo strumento sul materiale, oppure il cambiamento delle relazioni spaziali che l'attore del movimento aveva con il suo ambiente. Le etichette attribuite ai posti sono anch'esse frutto di un artificio pratico, esse indicano le precondizioni agli eventi successivi in termini di disponibilità di risorse (come strumenti o materiali), di accessibilità ai luoghi per effettuare movimenti, e di condizioni che permettono lo svolgersi di azioni che non si possono caratterizzare in termini di disponibilità o di accessibilità, ma con condizioni dell'attore quali ad esempio il non essere impegnato in altre attività.
Con *Disponibile(x)* si vuole indicare una condizione di disponibilità immediata della risorsa x, con *Accessibile(y)* si vuole indicare una condizione di Accessibilità a compiere un movimento verso il luogo y, con *Pronto* e *Occupato* si indicano le condizioni per l'azione da parte dell'attore.
Il fatto di esprimere una procedura tramite una rete nei soli termini di precondizioni ed eventi può apparire, ad un primo esame, una interpretazione del significato di condizione ed evento che va contro lo spirito originale del formalismo. La giustificazione all'uso che qui viene fatto delle reti è la seguente: il dizionario procedurale esprime le relazioni di dipendenza causale tra eventi di una procedura, quello che interessa quindi non è tanto specificare come le azioni svolte si ripercuotono sullo stato del mondo che viene costruito, (questo dipenderà in buona parte dalla situazione particolare in cui l'attore o gli attori della procedura sono immersi), piuttosto evidenziare il fatto che l'occorrere di un evento fa sì che possa accadere quello successivo, vale a dire che per togliersi le calze bisogna togliersi prima le scarpe e non importa se si è sul letto o in giardino, e che dopo, le scarpe saranno sul pavimento vicino al gatto.

4. RELAZIONI TRA GLI UNIVERSI

Sebbene non vi siano, almeno a livello di dizionario, correlazioni esplicite tra le entità rappresentate nei vari

universi, si può notare come sia facile trovare delle ridondanze. Infatti se ogni frase pronunciata si proietta nei tre universi perchè ognuno ne focalizza un aspetto ben preciso, allora è lecito supporre che di alcuni concetti sono presenti, in più di un dizionario, descrizioni da diversi punti di vista.
L'esempio più evidente è rappresentato dagli strumenti, che pur essendo entità dello spazio fisico in quanto occupano volume e posizioni nello spazio, trovano posto nel dizionario fisico; ma in qualità di oggetti che rendono possibili certe operazioni, e quindi, in qualità di risorse per compiere operazioni, devono essere rappresentati rispettivamente anche nel dizionario operativo e procedurale. Anche i materiali hanno una caratteristica simile.
Per quanto riguarda le operazioni ed i movimenti il discorso è solo un poco più complicato anche se sostanzialmente identico: le operazioni vengono compiute non senza effettuare qualche sorta di movimento, e viceversa ogni movimento viene compiuto con l'ausilio di qualche strumento (ed è quindi per questo motivo anche operazione), risulta allora giustificata la corrispondenza tra operazioni e movimenti negli universi fisico ed operativo. Resta comunque il fatto che alcuni movimenti caratteristici di operazioni non sono molto significativi, questo perchè non sono tali da modificare la scena fisica che il Multiversum è in grado di costruire; è allora la dimensione strumentale che ha maggior peso confinandoli (o meglio restringendoli) ad un solo universo. Questo è il caso di operazioni come *stringere un bullone* oppure *allacciarsi le scarpe*.
Infine, un'ultima parola riguardo al dizionario procedurale; le relazioni tra questo e gli altri due dizionari sono molteplici. Lì si esprimono le relazioni di dipendenza causale e di concorrenza tra eventi, eventi che sono ''catalogati'' negli altri dizionari perchè movimenti o operazioni (la distinzione ora dovrebbe essere chiara), nonchè condizioni di azioni, quali la disponibilità di strumenti e materiali (ecco un legame) e di luoghi in cui sono possibili movimenti (ed eccone un'altro).

5. ALCUNI ESEMPI

Segue un esempio di modellazione di un semplice ristorante. L'universo fisico è caratterizzato dai seguenti oggetti (Fig. 1):

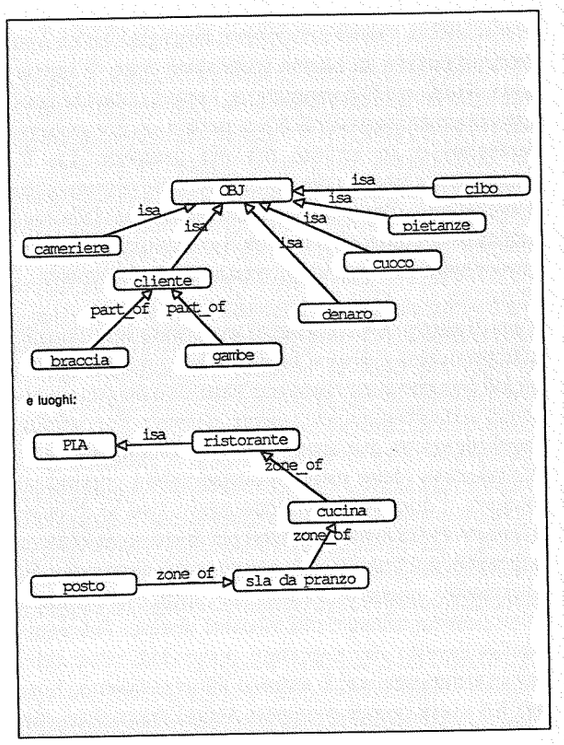

Fig. 1

Non sono state evidenziate tutte le relazioni per ragioni di brevità.
I movimenti sono: ENTRARE ed ANDARE, esisterà poi una descrizione di una scena iniziale del tipo:

[cameriere **IN** ristorante]
[cuoco **IN** ristorante]
[cibo **IN** cucina]
...

dove, a seconda del livello di dettaglio voluto, verranno specificate le relazioni spaziali di tutti gli oggetti tra loro, come tavoli, sedie, posate e stoviglie.

Dal punto di vista operativo si possono individuare alcuni strumenti quali gli arti degli attori, l'apparato uditivo del camerire e del cuoco, l'apparato vocale del cameriere e del cliente, la carta di credito del cliente etc...

Tutte le successioni possibili degli eventi (e quindi delle trasformazioni delle scene fisiche) sono espresse dalla seguente rete ad automi sovrapposti (Fig. 2):

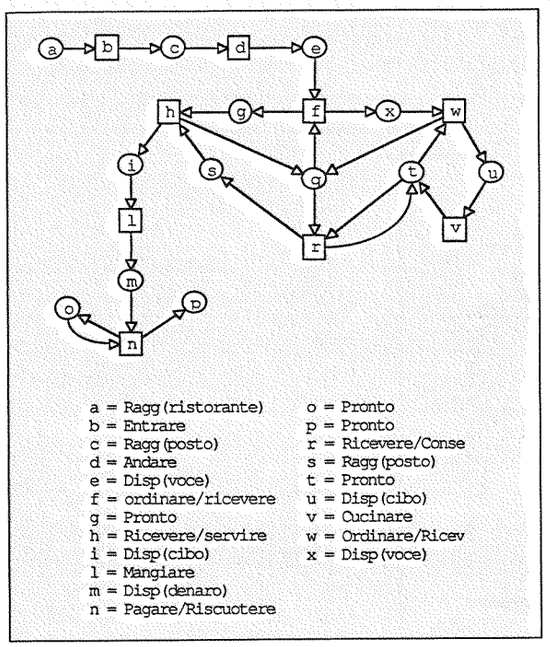

Fig. 2

dove (a,b,c,d,e,f,g,h,i,l,m,n,p) è la partizione relativa alla macchina a stati che modella il cliente, (f,x,w,q,r,s,h) è la partizione relativa al cameriere, (u,v,w,t,r) è la partizione relativa al cuoco ed (o,n,) la partizione di un eventuale cassiere. Il cliente si sincronizza con il camerire e con il cassiere rispettivamente in (f,h) e (n). Il cuoco ed il cameriere si sincronizzano in (w,r).
Le transizioni quali ''mangiare'' e ''cucinare'' possono essere ulteriormente raffinate.

6. IL LIVELLO "HIC ET NUNC"

È a livello ''Hic et Nunc'' che i legami esistenti tra gli universi divengono espliciti e che i concetti classificati nei dizionari si uniscono a formare una successione di ''scene'', di ''archivi'' e di ''marcature'' che determinano, rispettivamente, le evoluzioni di situazioni spaziali, delle disponibilità di materiali e di strumenti per compiere operazioni, degli eventi accaduti e dello stato degli attori di questi. All'Hic et Nunc apparterranno pertanto le 'scene': insiemi di formule (chiuse) derivanti dalla trascrizione in un linguaggio logico al prim'ordine dei concetti classificati come frames appartenenti alle tassonomie di luoghi, oggetti e movimenti del dizionario fisico, unite agli assiomi che caratterizzano le relazioni spaziali; gli 'archivi': insiemi di strumenti, operazioni e materiali classificati nel dizionario operativo; e le ''storie di esecuzione'', determinate dallo ''scattare'' delle transizioni e dall'evolvere delle 'marcature' dei sistemi SA che il dizionario procedurale classifica.
La **figura 4** vuole essere una esemplificazione di come istanze di oggetti classificati nei dizionari costituiscano, nell'Hic et Nunc, 'conoscenza' derivante dall'unione di informazioni provenienti da diversi domini. La parte superiore contiene delle rappresentazioni (semplificate) di oggetti del dizionario: il frame 'pagare', appartenente alla tassonomia di operazioni, che è quindi classificato nel dizionario operativo; il frame 'uscire', che appartiene alla tassonomia di movimenti ed è classificato nel dizionario fisico. La porzione di sistema-SA con le componenti 'cassiere' e 'cliente', che appartiene al dizionario procedurale. La parte inferiore contiene la successione di tre 'scene', 'archivi' e 'marcature' dove, nella componente relativa all'universo fisico il cambiamento è dovuto all'accadere dell'evento 'ESCE' (con soggetto il cliente); nella componente relativa all'universo operativo la modificazione che avviene è data dalla scomparsa del 'denaro' dall'archivio dei materiali disponibili (al cliente); ed infine, le modificazioni della componente procedurale, sono date dallo 'scattare' delle transizioni e dal conseguente ''avverarsi'' delle 'postcondizioni' ed ''invalidarsi'' delle 'precondizioni' delle transizioni coinvolte.

7. CONCLUSIONI

Nell'impostazione di un confronto con altri sistemi di rappresentazione, basati su ipotesi di lavoro più ''classiche'', il 'Multiversum' presenta analogie e differenze ed è interessante notare come partendo da un paradigma diverso si sia giunti ad avere l'esigenza di utilizzare metodologie di rappresentazione simili. In primo luogo la distinzione fra i tre universi del discorso (fisico, operativo e procedurale) potrebbe sembrare un artificio al fine di meglio esprimere conoscenza di carattere 'dichiarativo' e 'procedurale' [19]. Una affermazione di questo tipo non sarebbe precisa in quanto dimenticherebbe che è la priorità (logica) della dimensione pragmatica della comunicazione umana a generare, nella teoria del Multiversum, un mondo (di parole)

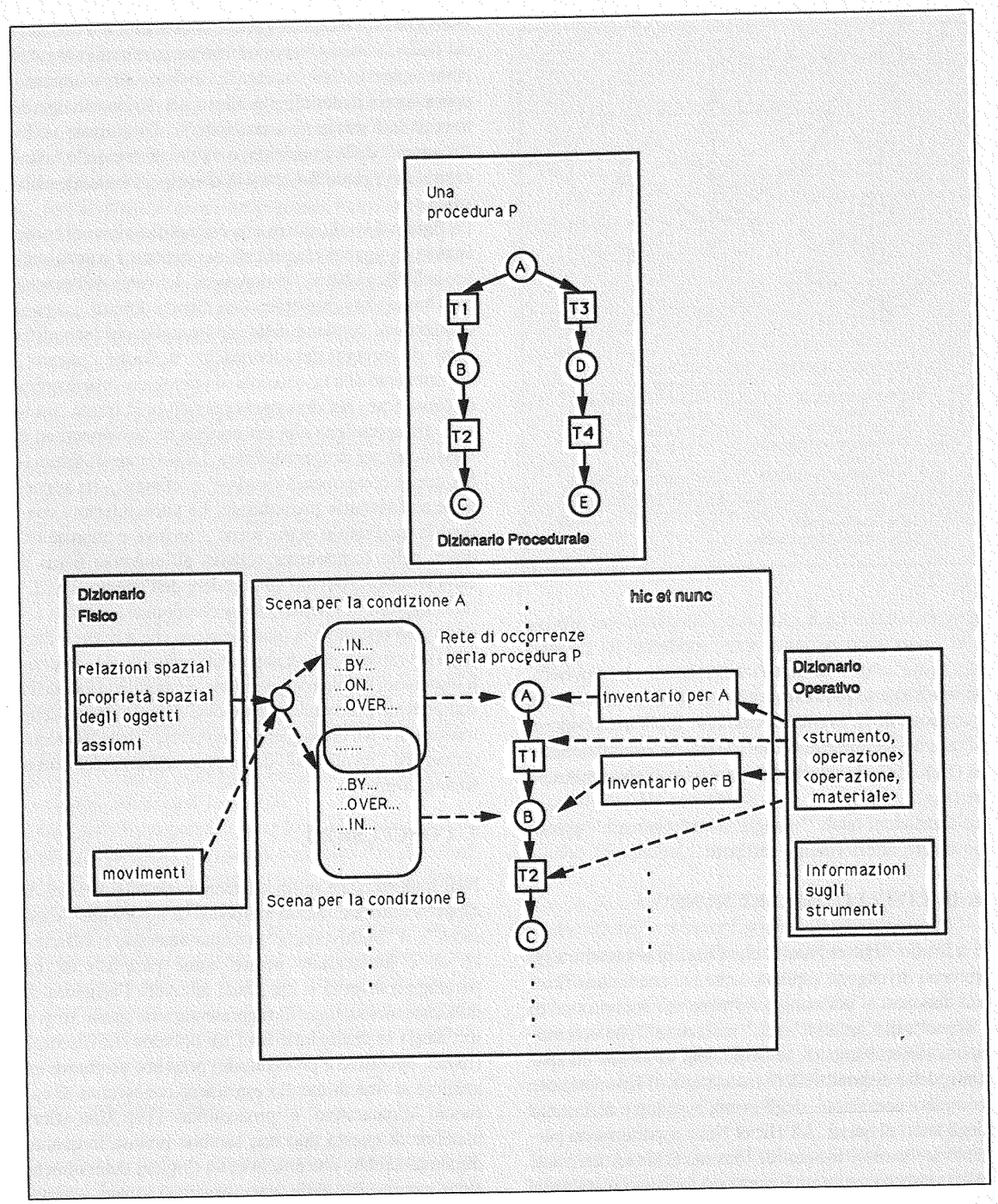

Fig. 3 La struttura del Multiversum.

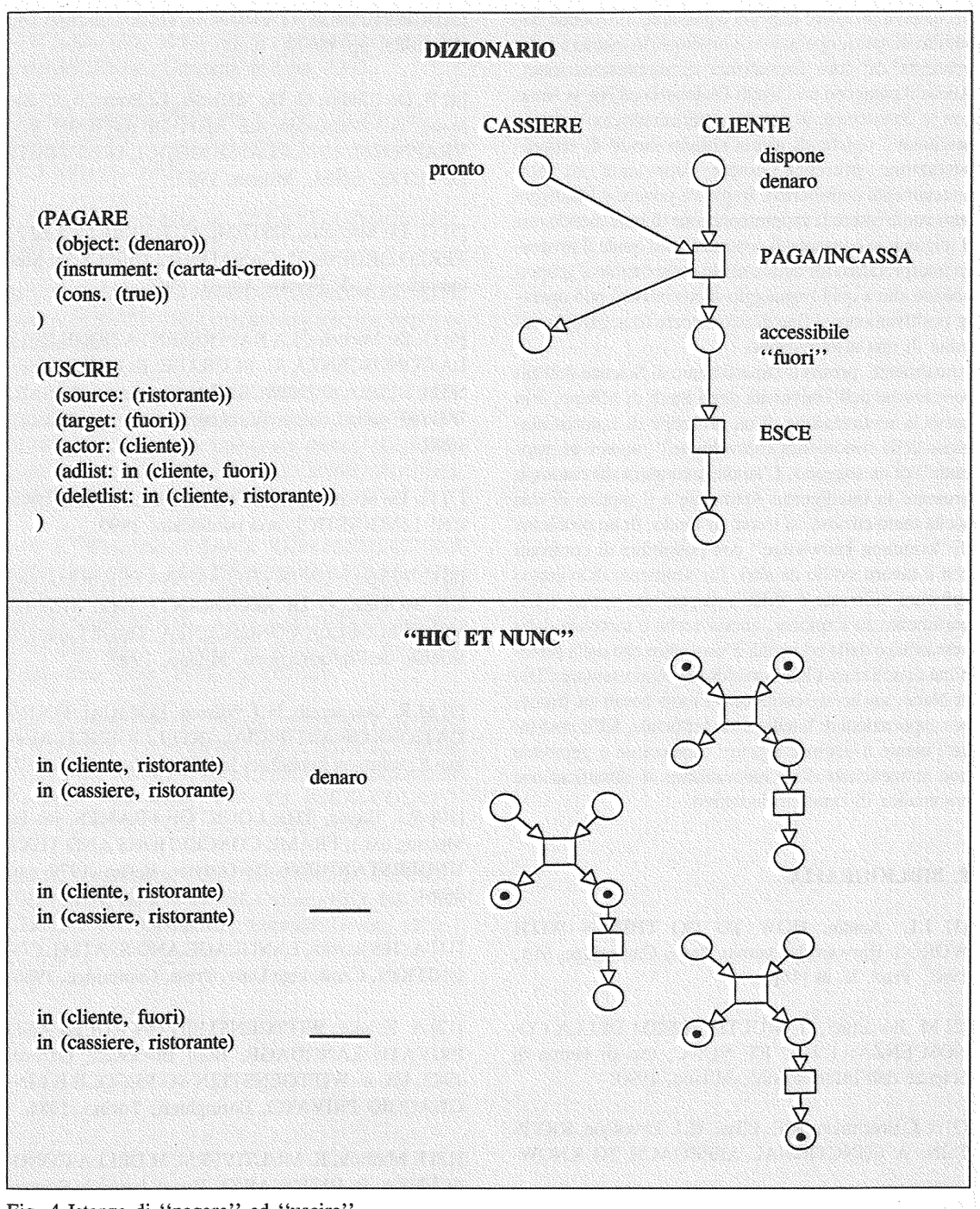

Fig. 4 Istanze di ''pagare'' ed ''uscire''.

nel quale il prendere impegni è possibile, un mondo costituito di spazi, operazioni e processi (in maniera indipendente dal loro formalismo di rappresentazione). Anche il paragone tra i livelli Dizionario ed Hic et Nunc con le conoscenze di carattere 'terminologico' ed 'asserzionale' tipiche di alcuni sistemi ibridi[4] di rappresentazione potrebbe essere fuorviante se non attentamente considerato. In parole povere il Multiversum vuole essere la rappresentazione di quel mondo che il linguaggio consente di costruire e sul quale il processo interpretativo (di ogni singolo componente il gruppo che intorno a quel linguaggio si costituisce) può operare positivamente al fine di riconoscere (o negare) la validità di una affermazione.
Attualmente, presso il Dipartimento di Scienze dell'Informazione dell'Università degli Studi di Milano, è in corso la realizzazione di un prototipo di rappresentazione della conoscenza coinvolta nell'''andare al ristorante'' di un soggetto. L'ampia letteratura sui ristoranti presente in Intelligenza Artificiale è il motivo di una scelta tanto curiosa, si tratta, in fondo, di un problema di ''common knowledge'' con possibilità di confronti con il lavoro svolto da altri. Lo strumento di sviluppo software utilizzato è 'KEE', disponibile presso il Dipartimento su Explorer, questa scelta è motivata dalla versatilità e dalla comodità d'uso, oltre che dalla possibilità di utilizzare i KEE-worlds per rappresentare l'Hic et Nunc; anche se, come tutti i tools basati su frames per applicazioni di Intelligenza Artificiale, KEE costringe l'utente a definire a priori tassonomie e gerarchie non ammettendo alcun meccanismo di classificazione automatica di carattere induttivo.

8. BIBLIOGRAFIA

[1] J.L. Austin, HOW TO DO THINGS WITH WORDS, Harward University Press, Cambridge, MA, 1962. Trad. it. in [16].

[2] M. Bonamici, IL MULTIVERSUM DELLA CONOSCENZA: L'HIC ET NUNC, tesi di laurea di Scienze dell'Informazione, Milano, 1990.

[3] R.J. Brachman, R.E. Fikes, H.J. Levesque, KRYPTON: A FUNCTIONAL APPROACH TO KNOWLEDGE REPRESENTATION, in: IEEE Computer **16**, (10) 1983, pp. 67-73.

[4] F. De Cindio, G. De Michelis, L. Pomello, C. Simone, A. Stragapede, LE RETI DI AUTOMI SOVRAPPOSTI: UNA CLASSE MODULARE DI RETI DI PETRI, ENEL, Milano, 1987.

[5] G. De Michelis, SISTEMI AUTOPOIETICI DEL TERZO ORDINE. IL CASO DEGLI UFFICI, in: Metamorfosi, I, **3**, Franco Angeli, 1986.

[6] G. De Michelis, LA RAPPRESENTAZIONE DELLA CONOSCENZA AL PLURALE: IL MULTIVERSUM DELLA CONOSCENZA, Quaderno IRRSAE, IRRSAE della Lombardia, in corso di stampa, Milano, 1990.

[7] G. De Michelis, SISTEMI SOCIALI COME GIOCHI LINGUISTICI, non pubblicato, 1990.

[8] C. Ferigato, UN INVITO A CENA PER SCHANK: UN MODELLO DI RISTORANTE NEL MULTIVERSUM DELLA CONOSCENZA, Tesi di Laurea in Scienze dell'Informazione, Milano, 1990.

[9] M.R. Genesereth, N.J. Nilsson, LOGICAL FOUNDATIONS OF ARTIFICIAL INTELLIGENCE, Morgan Kraufmann Publishers Inc., Los Altos, CA, 1987.

[10] P.J. Hayes, THE LOGIC OF FRAMES, in: D. Metzing (ed.), FRAME CONCEPTIONS AND TEXT UNDERSTANDING, De Gruyter, Berlin, 1979, pp. 46-61.

[11] A. Herskovitz, LANGUAGE AND SPATIAL COGNITION, Cambridge Univ. Press, Cambridge, 1986.

[12] S. Kripke, WITTGENSTEIN ON RULES AND PRIVATE LANGUAGE, Basil Blackwell, Oxford, 1982. Ed. it. WITTGENSTEIN SU REGOLE E LINGUAGGIO PRIVATO, Boringhieri, Torino, 1984.

[13] I. Maffioli, IL MULTIVERSUM DELLA CONOSCENZA: IL DIZIONARIO, Tesi di laurea in Scienze dell'Informazione, Milano, 1990.

(4) Ad esempio [3].

[14] H. Maturana F. Varela, EL ÁRBOL DEL CONOCIMIENTO, 1984. Ed. it. L'ALBERO DELLA CONOSCENZA, Garzanti, Milano, 1987.

[15] N.J. Nilsson, PRINCIPLES OF ARTIFICIAL INTELLIGENCE, Tioga Publishing Company, Palo Alto, California, 1980.

[16] M. Sbisà, (a cura di), GLI ATTI LINGUISTICI, Feltrinelli, Milano, 1978.

[17] D.S. Scott, CAPTURING CONCEPTS WITH DATA STRUCTURES, Preliminary Version, non pubblicato, 1986.

[18] J.R. Searle, MINDS, BRAINS AND PROGRAMS, in: THE BEHAVIORAL AND BRAIN SCIENCE, Cambridge University Press, Cambridge, 1980. Ed. it. MENTI CERVELLI E PROGRAMMI, CLUP, Milano, 1984.

[19] T. Winograd, FRAME REPRESENTATIONS AND THE DECLARATIVE/PROCEDURAL CONTROVERSY, in: D.G. Bobrow, A.M. Collins, (ed.) REPRESENTATION AND UNDERSTANDING: STUDIES IN COGNITIVE SCIENCE, Academic Press, New York, 1975, pp. 185-210.

[20] T. Winograd F. Flores, UNDERSTANDING COMPUTER AND COGNITION A NEW FOUNDATION FOR DESIGN, Ablex Publishing Corporation, Norwood, New Jersey, 1986. Ed. it. CALCOLATORI E CONOSCENZA, Mondadori, Milano, 1987.

[21] L. Wittgenstein, PHILOSOPHISCHE UNTERSUCHUNGEN, Basil Blackwell, Oxford, 1953. Ed. it. RICERCHE FILOSOFICHE, Einaudi, Torino, 1983.

A PROLOG SOLUTION TO CONSTRAINT SATISFACTION PROBLEMS

Massimo Paltrinieri (*)
Bull HN Italia
Research & Development Centre
Pregnana Milanese

ABSTRACT

A Prolog program that solves Constraint Satisfaction Problems (CSPs) embedding filtering algorithms in a common tree search structure is presented. Its modularity allows varying the filtering algorithms without affecting the tree search framework. Hence the program is a tool for determining the optimal amount of simplification to pursue at each node of the search.

RIASSUNTO

Questo lavoro presenta un programma Prolog che permette di risolvere i Problemi di Soddisfacimento di Vincoli inserendo diversi algoritmi di filtro in una struttura comune di ricerca su albero. La sua modularità consente di variare gli algoritmi di filtro, ferma restando la struttura di ricerca. Tale programma è quindi uno strumento per determinare la quantità ottimale di semplificazione da applicare ad ogni nodo della ricerca.

1. INTRODUCTION

Constraint Satisfaction Problems (CSPs) have received much attention due to the wide range of practical problems they can model. Examples are theorem proving [12], machine vision [15], event scheduling [13] and fleet routing [14].

(*) Current address: Stanford University, Department of Computer Science, Stanford, CA 94305, USA.

The components of a CSP are defined as follows:
- $V = \{v_1,v_2,\ldots,v_n\}$ a set of variables
- $D = \{D_1,D_2,\ldots,D_n\}$ the set of variable domains
- $C = \{C_{ij}C_{ij}$ is a subset of the Cartesian product $D_i \times D_j$ for some i,j in $1,\ldots,n\}$.

Only finite domains of discrete values and binary symmetric constraints are considered here. A CSP can be represented as a constraint graph G(V,E): V is the set of nodes, each with an associated domain and E contains an edge i-j iff C_{ij} is in C. Given V,D,C, solving a CSP means to find all the assignments of values in D to the variables in V such that the constraints in C are satisfied.
A way to perform it is by tree search + filtering, i.e. by deleting the values that cannot take part in any solution at each step of the search. Nadel [11] provides a unified framework of the form tree search + filtering to several constraint satisfaction algorithms. The modularity of the tree search + filtering procedure allows varying the filtering algorithms without affecting the tree search framework.
The filtering algorithms in [3,7,11] are subjected to the following drawbacks:
- they are expressed in pseudocode
- they assume some among the following restrictions:
  complete constraining: all variables constrain one another
  same domain size: all variables have the same domain size
  variable selection: variables are processed in a uniform order
  check selection: consistency checks are performed in a uniform order.

In this paper we give a Prolog program for the tree search + filtering process. The generality of the code we propose is such that the mentioned restrictions are not assumed.
As our program allows determining the best simplification algorithm to embed in the search framework, it is a practical tool to decide how much filtering is cost-effective to pursue during the search. The empirical algorithm comparison is achieved by changing the call to the filtering procedure in the main program. Due to its relational form, Prolog seems to be particularly suitable both to represent and to solve CSPs with a modest programming effort.
Recent work [1,4,5] has also been done to implement Constraint Logic Programming, an extension of Logic Programming in which the unification algorithm as used in Prolog is replaced by a more general operation called constraint satisfaction. Though these techniques are promising, they require to modify the Logic Programming interpreter or compiler to incorporate new features.

2. TREE SEARCH + FILTERING

A straightforward way to solve a CSP is by backtracking, a tree search procedure consisting of:

1 - given an instantiated variable, extending it with a new branch for each value in the domain of a new selected variable
2 - checking such values against the past variables along the corresponding branch of the tree
3 - recursively repeating the procedure for each value found consistent till all variables are instantiated.

Due to the trashing behaviour that often accompanies backtracking, consistency algorithms were developed in [15,10,6,9] to eliminate inconsistencies before they are discovered by the search. However, after the execution of the consistency algorithm, a non trivial search space may remain to be explored. Increasing the order of filtering reduces the search space but it may cost more than is saved by a less extensive search. In these problems a filtering algorithm embedded inside a tree search procedure may be more cost effective.
The tree search + filtering process consists of:

1 - instantiating a variable to a different value in its label, decomposing the problem into independent subproblems such that the union of all their solutions is the solution of the original problem
2 - applying consistency filtering to each subproblem, abandoning each of them whenever an inconsistency is detected
3 - recursively repeating till all the variables have been instantiated.

A mathematical analysis of the optimal amount of filtering to pursue in the tree search has not been developed yet. Empirical analysis is carried out by embedding different filtering algorithms in the tree search procedure and then by comparing their performances. Examples are in [8] for the graph isomorphism problem, in [3] for the N-queens puzzle and in [11] for the ''confused'' N-queens puzzle.

The program in Fig. 1 is a Prolog implementation of the tree search + filtering process. The procedure tree-search/6 is called with the following input values: Past is empty, Future is a set of variables, L is the initial labelling, Edges is a set of binary constraints, Filter is the name of the filtering algorithm and Solution is uninstantiated. After its execution, Solution is the set of assignments to variables of values in L simultaneously satisfying all the constraints in Edges.
Our implementation allows the embedding of different filtering algorithms without affecting the tree search framework. This task is now exemplified exploiting backward checking and forward checking as filtering algorithms.
We call backward checking a filtering algorithm which filters the present against the past, i.e. refines the label of the present variable to the values consistent with the past instantiations. Backtracking is supplied by the instantiation to 'bc' of the Filter variable in Fig. 1:
backtracking = tree-search + backward-checking.
The Prolog code of backward checking is shown in Fig. 2.

Forward checking is another filtering algorithm. Embed-

```
tree_search(Past,Future,L,Edges,Filter,Sols) :-
  nonempty(Future),
  select_var(Future,L,Edges,Present,NextFuture),
  label(Present,L,Label),
  subtrees_search(Label,Past,Present,NextFuture,L,Edges,Filter,Sols).

tree_search(_Past,Future,Sol,_Edges,_Filter,Sol) :-
  empty(Future).

subtrees_search(Label,Past,Present,Future,L,Edges,Filter,Sols) :-
  nonempty(Label),
  select_value(Label,Value,Rest),
  assign(Present,Value,L,SubL),
  filtering(Filter,Past,Present,Future,SubL,Edges,ReducedL),
  (  ReducedL \== inconsistent ->
       add(Present,Past,NewPast),
       tree_search(NewPast,Future,ReducedL,Edges,Filter,Sol)
  ;    empty(Sol)
  ),
  append(Sol,RestSols,Sols),
  subtrees_search(Rest,Past,Present,Future,L,Edges,Filter,RestSols).

subtrees_search(Label,_Past,_Present,_Future,_L,_Edges,_Filter,Sols) :-
  empty(Label),
  empty(Sols).
```

Fig. 1

```
filtering(bc,Past,Present,_Future,L,Edges,FilteredL) :-
     ancestors(Present,Edges,Ancestors),
     intersection(Ancestors,Past,PastAncestors),
     revise(PastAncestors,Present,L,FilteredL).

filtering(fc,_Past,Present,Future,L,Edges,FilteredL) :-
     offsprings(Present,Edges,Offs),
     intersection(Offs,Future,FutureOffs),
     revise(FutureOffs,Present,L,FilteredL).

revise(Vars,Present,L,FilteredL) :-
     assignment(Present,L,Value),
     select_check_var(Vars,X,Rest),
     label(X,L,LX),
     refine(Present,Value,X,LX,RefinedLX),
     (  nonempty(RefinedLX) ->
          new_label(X,RefinedLX,L,TmpL),
          revise(Rest,Present,TmpL,FilteredL)
     ;  empty(RefinedLX) ->
          FilteredL = inconsistent
     ).
revise(Vars,_Present,FilteredL,FilteredL) :-
   empty(Vars).

refine(Var1,Value1,Var2,Label,Refined) :-
     select_value(Label,Value2,Rest),
     (  inequality(Var1,Value1,Var2,Value2) ->
          add(Value2,RefinedRest,Refined),
          refine(Var1,Value1,Var2,Rest,RefinedRest)
     ;    refine(Var1,Value1,Var2,Rest,Refined)
     ).
   refine(_Var1,_Value1,_Var2,Empty,Empty) :-
     empty(Empty).
```

Fig. 2

ded in the tree search + filtering procedure, it solves several CSPs most efficiently [3,8,11]. The Prolog code of the forward checking algorithm is also shown in Fig.2. In order to embed it in tree-search, the Filter variable in the call to filtering in Fig. 1 is to be instantiated to 'fc'. Both backtracking and forward checking consist of a top level selecting the variables to be checked against the present one and of a common partial arc consistency algorithm performig the filtering.

3. GENERALITY

Several papers [2,3,6,7,8,9,11,15] formally describe filtering algorithms in pseudo-code since their translation into conventional programming languages would result in complex programs inadequate to the algorithm comprehension. The closer the description is to a programming language, the more the algorithm generality is reduced to mantain readibility. We state that Prolog brings a great improvement to the problem. Consider for example the Pascal-like version of backtracking and forward checking proposed in [11] and reprinted in fig. 3. The procedures are quite close to Pascal but the complete constraining, variable selection and check selection assumptions are introduced. They simplify the algorithms in the following way:

- *complete constraining:*
since all variables constrain one another, the variables to check against the present variable (backtracking) and against the most recent past variable (forward checking) are simply determined by the FOR loops at lines 12 (backtracking) and 27 (forward checking)

- *variable selection:*
since variables are processed in a uniform order, the new variable is simply determined by incrementing the present variable in the recursive call at lines 7 (backtracking) and 21 (forward checking)

- *check selection:*
since the consistency checks are performed in uniform order, the values in the labels of the checking variables are simply determined by the FOR loops at lines 32 and 35.

Our formulation is not subjected to these assumptions:


```
1    PROCEDURE BT(k,d);
2    Check_Backward(k,d,empty_domain);
3    IF not empty_domain THEN
4            FOR zk <- each element of Copy(d[k]) DO
5                    BEGIN
6                    d[k] <- z(k);
7                    IF k=n THEN output(d) ELSE BT(k+1,d)
8                    END
9    END;

10   PROCEDURE Check_Backward(k,VAR d,VAR empty_domain);
11   empty_domain <- False;
12   FOR p <- 1 TO k-1 WHILE not empty domain DO
13                   revise(k,p,d,empty_domain,dummy)
14   END;

15   PROCEDURE FC(k,d);
16   Check_Forward(k,d,empty_domain);
17   IF not empty_domain THEN
18           FOR zk <- each element of Copy(d[k]) DO
19           BEGIN
20           d[k] <- (zk);
21           IF k=n THEN output(d) ELSE FC(k+1,d)
22           END
23   END;

24   PROCEDURE Check_Forward(k,VAR d, VAR empty_domain);
25   empty_domain <- false;
26   IF k>1 THEN
27           FOR f <- k TO n WHILE not empty_domain DO
28                   revise(f,k-1,d,empty_domain,dummy)
29   END;

30   PROCEDURE revise(i,j,VAR d,VAR empty_domain,VAR change);
31   change <- False;
32   FOR vali <- each element of Copy(d[i]) DO
33           BEGIN
34           support_found_for_vali <- False;
35           FOR valj <- each element of d[j]
36                   WHILE not support_found_for_vali DO
37                           IF check(i,vali,j,valj)
38                                   THEN support_found_for_vali <- True;
39           IF not support_found_for_vali THEN    BEGIN
40                                                 d[i] <- d[i] - (vali);
41                                                 change <- TRUE
42                                                 END
43           END;
44   IF d[i] = empty THEN empty_domain <- True
45   END;
```

Fig. 3

- *complete constraining:*
the constraint graph is not implicit in the specialized algorithm but is an input variable which can be instantiated to arbitrary sets of constraints

- *variable selection:*
the order in which variables are selected for instantiation is not fixed; several selection strategies can be implemented in the procedure selectvar of Fig. 1, exploiting the labelling and constraint graph structures. The general principle of trying to fail first can be employed to dynamically choose the optimal order. Instances of this general principle to variable selection are:

- select the variable participating in the highest number of constraints
- select the variable most constrained to the instantiated variables
- select the variable with the smallest label

- *check selection:*
the order in which the consistency checks are performed, determined by the procedure select-check in Fig. 2, can be optimized by similar heuristics such as:

- check the present variable against the variables whose label is least likely to succeed

- *same domain size:*
also this assumption, introduced e.g. in [7], is discharged because the labels are not restricted to have the same size.

4. EXAMPLES

We describe now the application of our formulation to the N-queens puzzle and to the graph coloring problem. The n-queens puzzle is stated as follows: N queens must be placed on a chessboard in such a way that no queen attacks any other queen. Two queens attack each other when they are on the same horizontal, vertical or diagonal line. Placing each queen on a different column and indexing columns and rows by positive integers, the N-queen puzzle can be formulated as a CSP where the problem components are (N=4):

- V = $\{1,2,3,4\}$
- D = $\{D_1=\{1,2,3,4\}, D_2=\{1,2,3,4\}, D_3=\{1,2,3,4\}, D_4=\{1,2,3,4\}\}$
- C = $\{C_{12},C_{13},C_{14},C_{23},C_{24},C_{34}\}$ s.t. $C_{ij}(x,y)$ <=> noattack(i,x,j,y)

The consistency check code for the N-queens puzzle is shown in Fig. 5a. A more general example of CSP is the graph coloring problem: colors taken from a finite set must be assigned to the vertices of a graph in such a way that vertices joined by an edge have different colors. An example is shown in Fig. 4. The components of this problem are:

- V = $\{1,2,3,4\}$
- D = $\{D_1=\{r,w\}, D_2=\{g,r,w\}, D_3=\{g,r,w\}, D_4=\{r,w\}\}$
- C = $\{C_{12},C_{13},C_{23},C_{24},C_{34}\}$ s.t. let x,y be two values in D_i and D_j respectively, then for each i,j $C_{ij}(x,y)$ <=> x=y.

Tree search + filtering solves the graph coloring problem using the consistency check code shown in Fig. 5b.

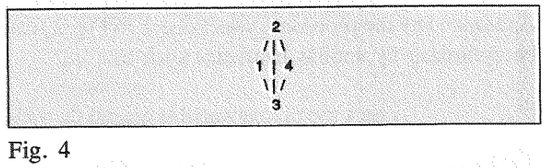

Fig. 4

```
a)      noattack(I,X,J,Y) :-
                X \== Y,
                K is J - I,
                DK is I - J,
                Z is Y - X,
                K \== Z,
                DK \== Z.

b)      inequality(_Var1,Value1,_Var2,Value2) :-
                Value1 \== Value2.
```

Fig. 5

5. CONCLUSIONS

A Prolog program finding all the solutions to CSPs has been presented. The labelling and constraint graph structures are exploited to discharge assumptions of previous papers such as complete constraining, same domain size, uniform variable order and uniform check order. Hence two drawbacks of previously described procedures are removed by our formulation since:
- it is executable
- it is more general. .

Characteristic features of the Prolog tree search + filtering procedure are:
- modularity: each of the modules, such as the filtering algorithm and the selection strategies, can be developed independently without affecting the rest of the program
- data abstraction: there is no visibility of any data structure representing a problem component
- clarity: constraints are expressed in a very declarative style
- flexibility: tree serach + filtering procedure is applicable to different CSPs by simply changing the consistency check code.

APPENDIX - DATA STRUCTURES AND UTILITIES

The Prolog code shown in previous figures is not subjected to any data structures. In this appendix we show:
- a possible implementation of the problem structures
- the instantiation of the input variables for the examples in section 5
- the code of the low level procedures on the basis of these choices (Fig. 6).


```
empty([]).

nonempty(X) :-
        X \== [].

member(Element,[Element|_]).
member(Element,[_|Rest]) :-
        member(Element,Rest).

memberchk(Element,[Element|_]) :- !.
memberchk(Element,[_|Rest]) :-
        memberchk(Element,Rest).

append([],L,L).
append([H|T],L,[H|R]) :-
        append(T,L,R).

add(X,L,[X|L]).

intersection([],_,[]).
intersection([Element|Elements],Set,Intersection) :-
        memberchk(Element,Set),
        !,
        Intersection = [Element|Rest],
        intersection(Elements,Set,Rest).
intersection([_|Elements],Set,Intersection) :-
        intersection(Elements,Set,Intersection).

replace(_Old,_New,[],[]).
replace(Old,New,[Old|Rest],[New|Rest]) :- !.
replace(Old,New,[H|Rest],[H|Rest1]) :-
        !,
        replace(Old,New,Rest,Rest1).

% labelling utilities

label(Var,Labelling,Label) :-
        member(Var:Label,Labelling).

new_label(Var,NewLabel,Labelling,NewLabelling) :-
        replace(Var:_OldLabel,Var:NewLabel,Labelling,NewLabelling).

assign(Var,Value,Labelling,NewLabelling) :-
        replace(Var:_OldLabel,Var:[Value],Labelling,NewLabelling).

assignment(Var,Labelling,Value) :-
        member(Var:[Value],Labelling).

% graph utilities

offsprings(Node,Graph,Offs) :-
        member(Node-Offs,Graph).

ancestors(Node,[X-Off|Graph],[X|Ancs]) :-
        memberchk(Node,Offs),
        !,
        ancestors(Node,Graph,Ancs).
ancestors(Node,[_XOffs|Graph],Ancs) :-
        ancestors(Node,Graph,Ancs).
ancestors(_Node,[],[]).


% selection utilities

select_var([X|Xs],_L,_Edges,X,Xs).

select_value([X|Xs],X,Xs).

select_check_var([X|Xs],X,Xs).
```

Fig. 6

- A possible implementation

- Past: a list of variables [VarVars]
- Future: a list of variables [VarVars]
- Labelling: a list of pairs variable:label [Var:Label-Labelling]
- Label: a list of values [ValueValues]
- Edges: a list of vertex-neighbours pairs [Vertex-NeighboursEdges] where Neighbours is a list of variables
- Solutions: a list of assigments [Var:ValueAssignments]
- Input variables

4-queens

- Past = []
- Future = [1,2,3,4]
- L = [1:[1,2,3,4],2:[1,2,3,4],3:[1,2,3,4],4:[1,2,3,4]]
- Edges = [1-[2,3,4], 2-[3,4], 3-[4], 4-[]]
- Filter = 'bc' or 'fc'
- Solutions = uninstantiated

Graph coloring

- Past = []
- Future = [1,2,3,4]
- L = [1:[r,w],2:[g,r,w],3:[g,r,w],4:[r,w]]
- Edges = [1-[2,3],2-[3,4],3-[4],4-[]]
- Filter = 'bc' or 'fc'
- Solutions = uninstantiated

ACKNOWLEDGMENTS

I am indebted to F. Torquati and A. Momigliano for useful discussions on this topic.

REFERENCES

[1] Colmerauer A., AN INTRODUCTION TO PROLOG III, Communications of the ACM, vol. 33, N. 7, July 1990.

[2] Davis E., CONSTRAINT PROPAGATION WITH INTERVAL LABELS, Artificial Intelligence, vol. 32, 1987, pp. 281-331.

[3] Haralick R.M. and Elliott G.L., INCREASING TREE SEARCH EFFICIENCY FOR CONSTRAINT

SATISFACTION PROBLEMS, Artificial Intelligence, vol. 14, 1980, pp. 263-313.

[4] Hentenryck P.V., CONSTRAINT SATISFACTION IN LOGIC PROGRAMMING, MIT Press, Cambridge, Mass., 1989.

[5] Jaffar J. and Lassez J.L., CONSTRAINT LOGIC PROGRAMMING, Proc. of the Fourteenth ACM Symposium of the Principles of Programming Languages, Munich 1987, pp. 111-119.

[6] Mackworth A.K., CONSISTENCY IN NETWORKS OF RELATIONS, Artificial Intelligence, vol. 8, 1977, pp. 99-118.

[7] Mackworth A.K. and Freuder E.C., THE COMPLEXITY OF SOME POLYNOMIAL NETWORK CONSISTENCY ALGORITHMS FOR CONSTRAINT SATISFACTION PROBLEMS, Artificial Intelligence, vol. 25, 1985, pp. 65-74.

[8] McGregor J.J., RELATIONAL CONSISTENCY ALGORITHMS AND THEIR APPLICATION IN FINDING SUBGRAPH AND GRAPH ISOMORPHISM, Information Sciences, vol. 19, 1979, pp. 229-250.

[9] Mohr R. and Henderson T.C., ARC AND PATH CONSISTENCY REVISITED, Artificial Intelligence, vol. 28, 1986, pp. 225-233.

[10] Montanari U., NETWORKS OF CONSTRAINTS: FUNDAMENTAL PROPERTIES AND APPLICATIONS TO PICTURE PROCESSING, Information Sciences, vol. 7, 1974, pp. 95-132.

[11] Nadel B.A., TREE SEARCH AND ARC CONSISTENCY IN CONSTRAINT SATISFACTION ALGORITHMS, in L. Kanal and V. Kumar (Eds), Search in Artificial Intelligence, Springer-Verlag, 1988.

[12] Purdom P.W.Jr and Brown C., AN AVERAGE TIME ANALYSIS OF BACKTRACKING ALGORITHMS, SIAM J. Comput., vol. 10, pp. 583-593.

[13] Rit J., PROPAGATING TEMPORAL CONSTRAINTS FOR SCHEDULING, Proc. Fifth National Conference on Artificial Intelligence (AAAI), Philadelphia, Pennsylvania, August 1986.

[14] Torquati F., Paltrinieri M., Momigliano A., A CONSTRAINT SATISFACTION APPROACH TO OPERATIVE MANAGEMENT OF AIRCRAFT ROUTING, Proc. Third Int. Conf. on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems, Charleston SC, 1990.

[15] Waltz D., UNDERSTANDING LINE DRAWINGS OF SCENES WITH SHADOWS, in P.Winston (Ed), The Psycology of Computer Vision, McGraw-Hill, 1975.

SEDAL: AN EXPERT SYSTEM FOR CLINICAL LINEAR ACCELERATORS

S. Bandini, G. Lavelli, S. Scamuzzo, G. Scielzo(*)
Dipartimento di Scienze dell'Informazione - Università di Milano
(*) Istituto di Fisica Sanitaria - Ospedali Galliera (Genova)

RIASSUNTO

In questo lavoro viene presentato SEDAL, un Sistema Esperto per la diagnostica di guasti di acceleratori lineari clinici. Verrano messe in risalto le caratteristiche principali del modello della conoscenza costruito e della classificazione euristica adottata come metodo di acquisizione e modellazione.

ABSTRACT

The basic features of the mainterrance knowledge modelling and the application of heuristic classification in diagnostic tasks are illustrated here in order to explain the structure of SEDAL, an implemented Expert System for clinical linear accelerators maintenance.

1. INTRODUCTION

In this paper we present the experience gained in the study and development of an Expert System for the maintenance of *clinical linear accelerators*. In this direction SEDAL (Sistema Esperto Degli Acceleratori Lineari), an Expert System for the monitoring on-line of most significant physical parameters and for faults diagnosis, has been realized.
The design and the implementation of this system required a careful analysis of the knowledge involved because of the great complexity of clinical linear accelerator (Clinac -4 [3]) installed at the Institute of ''Fisica Sanitaria - Ospedali Galliera'' in Genoa. The interaction with the maintenance expert has been guided by the application of methodological criteria we had selected in Expert Systems [5] research literature: the particular field identified and the features of the diagnostic tasks involved for this class of plant suggested different ways of modelling the expert knowledge [4,13]. The *structural* and the *decisional* models have been studied and the diagnostic part of the maintenance process has been approached thanks to the application of *heuristic classification* principles [2].

This paper will explain the basic choices taken to obtain the two kinds of modelling mentioned and how we adopted heuristic classification in our domain. The next section will contain some schematic information about this domain and further on the problems involved in the maintenance of a clinical linear accelerator will be briefly illustrated.

1.2 The domain

Clinical linear accelerators are complex systems which, accelerating charged particles in a linear way, produces radiation beams to be used for therapeutic purposes [6]. Radiation is produced when electrons flowing with a speed very close to c (c = light speed) strikes a tungsten target.

High energy X rays therapies are used in the treatment of cancers, taking advantage of the stronger radiation sensibility of tumorous cells with respect to healthy ones [8].

The analysis of problems related to the use of linear accelerators in the medical field underlines the need of a high level of precision in dosimetric factors accounts [9]. A satisfactory result of the application of this therapeutiac activities depends on the degree of accuracy employed, with respect to the exposure time and the beam intensity. A very important role is played by the use of a support for fault diagnosis in ordinary maintenance.

2. KNOWLEDGE ACQUISITION AND ANALYSIS

The knowledge acquisition stage, for building the knowledge base of our expert system, started with the study of the architecture and the functional principles of the accelerator [4], followed by the stage of radiation production process analysis.
Firstly, we obtained a functional model based on interconnected components: each component has been represented thanks to a black box model. This choice allowed the interconnection between the elements of the system to be noted. The consequence of this is that components modelling respect the locality principle: a fault in a component causes the propagation of anomalous values to the neighbours.

Secondly, the decisional process followed by experts in problem solving for diagnostic tasks has been analyzed.
Faults diagnosis can logically be divided into two steps (fig. 1): fault area localization and topological refinement.

After the malfunctioning has been noted, the expert links the data interpretation (from the console of the accelerator) to the subsystem of the entire plant where the fault could probably be present. These kinds of links are often based on heuristic considerations: only expert skill allows causal steps to be passed over and in this way it is possible to obtain an efficient solution (e.g. in terms of time) but also a loss of certainty. A consistency control can be made on the results obtained in this stage

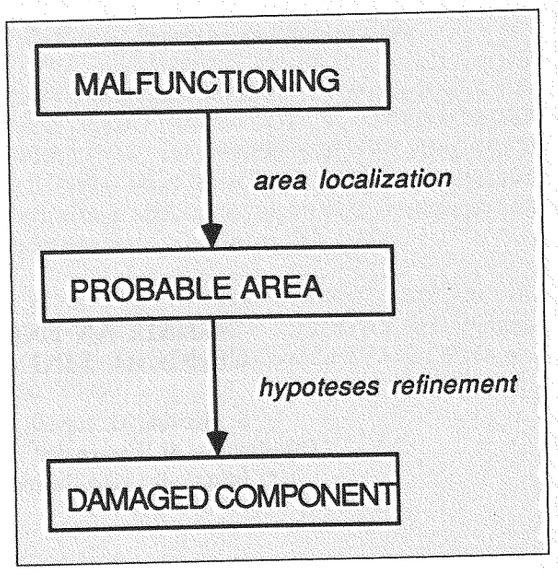

Fig. 1

thanks to particular measurements on predefined check points in the accelerator, in order to state which is the particular damaged element in the fault area.

The relevance of the first stage is based on narrowing down the inquiry from some hundreds to less than ten fault hypothesis.

In order to recognize and formalize hieratchies and causality related to diagnostic problems resolution, we followed the Knowledge Level Analysis: a methodology for problem analysis which can be indipendently applicated from the representation language employed and from the application [2].

The behavioural rules of the expert have been described using three types of assertions: terminological, relational and inferential.
With **terminological assertions** we wanted to unequivocally fix the meaning given by the expert to some terms (e.g., terms as ''hypothesis'', ''observable datum'', or '' symptom'', etc.)
The **relational assertions** indicate relationships and hierarchies among the concepts employed in diagnosis process. For instance, the relationship of co-presence between symptoms and hierarchies of fault hypoteses have been formalized.

With **inferential assertion** we characterized the inferential process employed by experts in diagnostic activity.
We recognized the abduptive nature of diagnosis which, by definition, tries to find causes (the fault) starting from consequences (observed anomalies).
The behavoiural model we found has been obtained through Heuristic Classification [clancey] criteria.
The choice of this methodology was motivated by the fundamental features of our problem, for instance the possibility of using pre-enumerated solutions (for each possible fault) and to articulate resolution process in three sequential and distinct computational phases:

- observable data abstraction
- heuristic mapping in a hierarchy of pre-enumerated solutions
- hierarchy refinement

These features correspond to ''requirements'' needed for the Heuristic Classification applicability for the design of Expert Systems (figg.2,3).

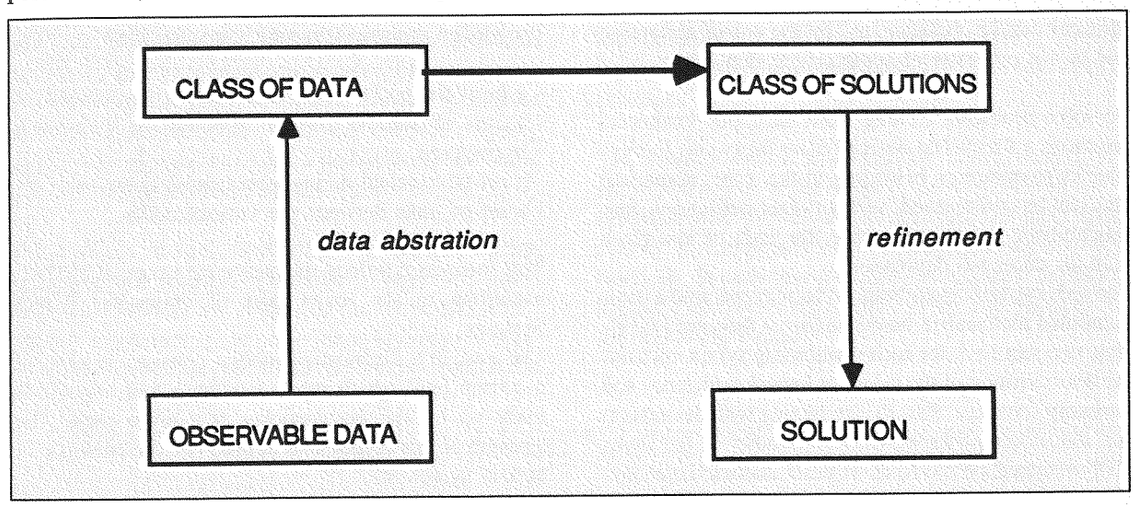

Fig. 2

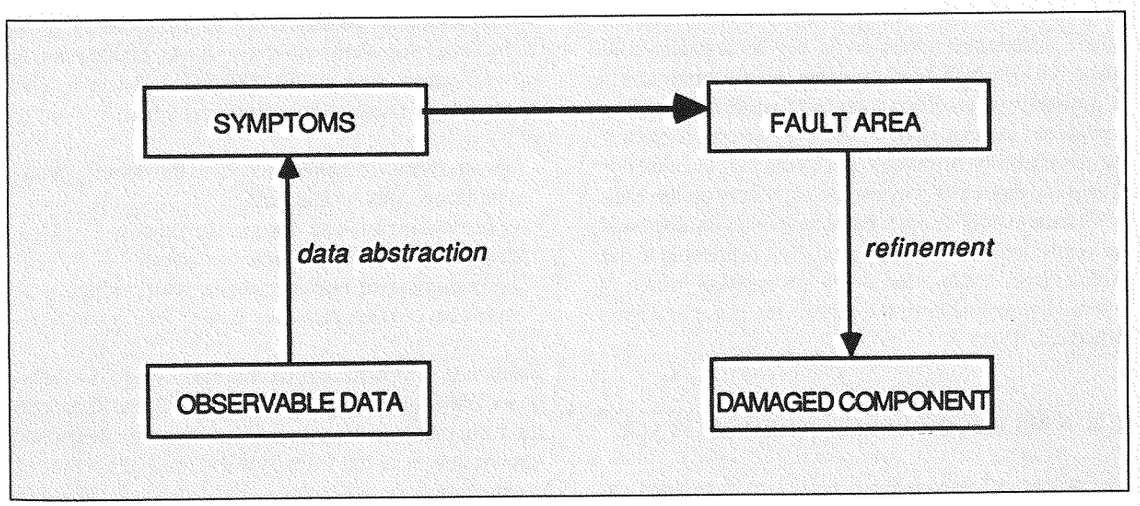

Fig. 3

3. THE ROLE OF UNCERTAIN KNOWLEDGE

In the design and realization of SEDAL we needed to emulate the expert's skill in taking decisions in an environment where information is incomplete and inexact. In fact, some useful data in diagnostic activity is not available, because it's acquisition is dangerous or too expensive in terms of time. Besides, data can be inexact because of the measurement tools employed for the survey. To these factors, connected to data nature, we must the add the factors related to types of inferences made by the expert for faults finding. These inferences can be characterized by the use of abduption and of the omission of some causal step.

To solve problems coming from the input weakness, we chose a formalism derived from fuzzy sets theory. Thanks to operators belonging to this type, numerical data can be transformed, without other definitions, into probabilistic values, mirroring the truth of assertions with an uncertain denotation.
For updating fault hypothesis probability we used a computational mechanism based on Bayes theorem [7,11]. This step required, for each relationship symptom/fault, the determination of the values of logical sufficiency and necessity (obtained directly interacting with the expert) and the values of the a priori probability of the faults [10] (assigned on the basis of exact statistic investigations made by a well-known producer of clinical linear accelerators).

Linear accelerator faults could not be considered as events yielded by a casual process; so the employment of probabilistic assertion to express uncertainty can be considered inexact [14]. In fact, arguments regarding the applicability of inverse probability in the description of the degree of a hypothesis certainty on the basis of a casual model of observed data have been discussed for years. In a bayesian perspective, it is possible to use probabilistic notions for every proposition which is potentially verifiable in the future, for instance a fault hypothesis [1].

4. KNOWLEDGE REPRESENTATION STAGE

The process of knowledge acquisition, developed using concepts from the Heuristic Classification methodology, has provided a set of associations between faults and symptoms, viewed as qualitative descriptions of observable data.
This set really underlines an associational causal model, which constitutes the starting point of the process of knowledge representation which will realize the translation of every kind of knowledge contained in our model to computational forms.
The actual goal of this process is to provide a knowledge network, as suggested by Savoir [12] development tool, chosen for the development of our Expert System.
This knowledge network is composed of a set of knowledge structures in which we can find:

- a head (or goal) which represents an hypothesis;
- a set of internal nodes representing antecedent hypotheses;
- a set of terminal nodes representing questions;
- a set of links between the various nodes.

This formalism reflects the backward nature of SEDAL according to the major part of diagnostic Expert Systems.
The goal in a diagnostic problem consists, in fact, of a certain fault which represents the actual hypothesis made up by the diagnostician at a given time. The strength of this Hypothesis depends on the presence of certain symptoms.

In the development of the process of knowledge representation we defined a proper methodology made up by collecting elements and concepts extabilished in various systems such as PROSPECTOR [15] et al. This methodology contains five principal steps:

- representation of main concepts of the domain (fault, symptroms, observable data);
- representation of data abstraction process;
- identification of hierarchical structures;
- representation of fault/symptoms relationships;
- evaluation of some particular probabilistic parameters.

Faults and symptoms are represented using probabilistic variables (assuming values between 0 and 1) reflecting from one side the guessing nature of the disgnostic science and from the other side the need of keeping in mind possible mistakes due to imprecision in measurements and in establishing particular concepts.

Observable data are indeed represented by questions contained in terminal nodes, which can be seen as the actual input of the system. The objective of the system and the problem characteristics have suggested a kind of mixed input: some data is obtained from external devices collected to the linear accelerator, while the others are explicitally requested from the user in an interactive dialogue.

The concepts of the symptoms are gained by making a process of data abstraction. Such a process can be represented in two ways: if data is obtained by answering ''yes-no'' questions, the corresponding symptoms are represented by probabilistic variables (assuming only the 0 and 1 values); when numeric data is represented introducing probabilistic distribution function mapping a particular physical value into a probabilistic one stored in the variable representing that symptom (see fig.4)

The analysis of each fault-symptoms association has highlighted the presence of common symptoms in several faults. This fact has suggested the introduction of a concept representing the copresence of common symptoms (syndrome) and of a fault taxonomy made up on the basis of these syndromes. Every syndrome contains symptoms built starting from data obtained just looking at the console of the Clinac-4, then makes a relation with a particular subsystem of the search space and gives the possibility of partitioning the search space simplifying and improving the search mechanism.

The symptom-syndrome relationships have a necessary nature in the sense that the presence of all symptoms collected in a syndrome is mandatory. In this case the representation of these relationships can be made using fuzzy logical relations which contitues the probabilistic translation of logical operators such as AND, OR and NOT. Obviously any combination of these operators is possible.

The symptoms-fault relationships reflect, indeed, the subjective nature of the heuristic association built up by the diagnostic experts and in this case the most satisfactory representation seems to be that using bayesian relations. In particular the implementation of the Bayes theorem that we choose requires the establishment of the following values:

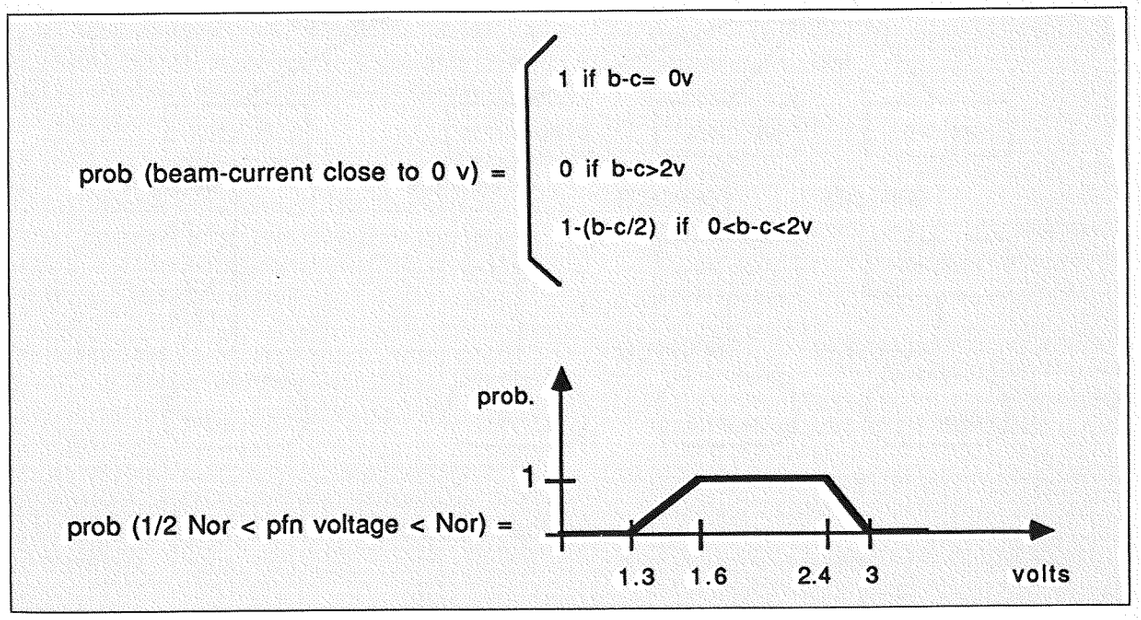

Fig. 4

- Prior probabilities of any fault, i.e. the probability accepted in the case of absence of any evidence;

- Logical sufficiency values LS = P(E/H) / P(E/H) is a measure of how much the presence of evidence E strenghtens the hypothesis H;

- Logical necessity value LN = P(E/H) / P(E/H) is a measure of how much the absence of evidence E makes the hypothesis H less likely.

The progressive updating of probabilistic values is made using the formulas:

odd (H/E) = LS * odd(H)
odd (H/E) = LN * odd H

The next figure shows an example of the knowledge structure referring to a particular symptom-fault association and inserted in the whole knowledge network.

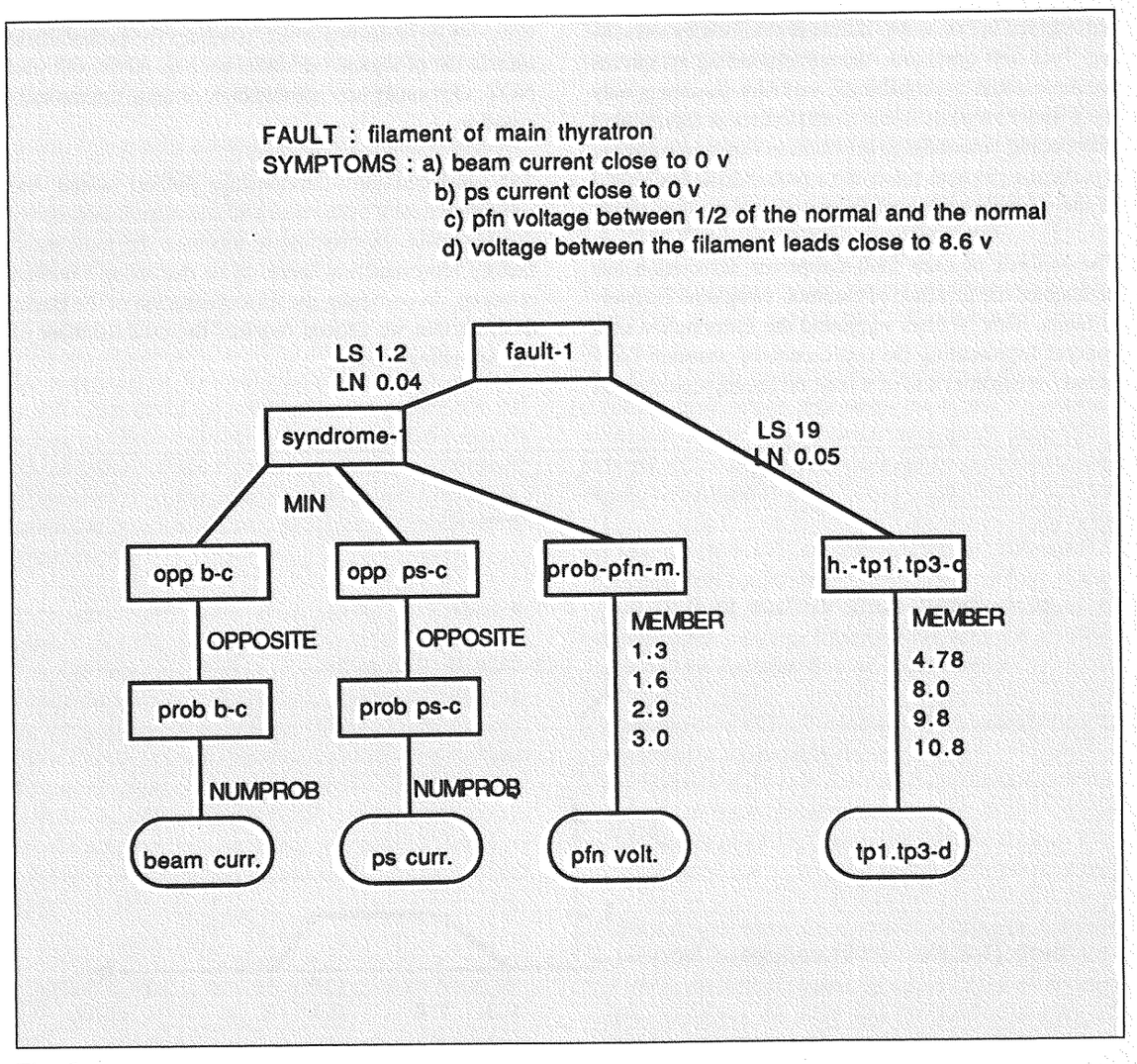

Fig. 5

5. KNOWLEDGE BASE ARCHITECTURE AND SEARCHING STRATEGIES

The whole knowledge network, composed of the various knowledge structures designed for every fault-symptoms association, presents a definite and useful hierarchical structure (see fig. 6).
In particular we can define three levels and discover inside some of them other specific sublevels:

- the diagnostic categories include variables which represent classes of faults and the faults themselves;
- pathological states describe various possible states of abnormal conditions or symptoms. These symptoms can be classified as primary symptoms (or syndromes) and secondary symptoms;
- observable data represents the evidence collected from the machine.

Observing fig.6 we can note that the arcs leaving classes nodes reach the syndrome nodes through the faults nodes. This suggests the possibility of directly linking classes to syndromes. The introduced link presents two different semantics:
. the class-syndrome link represents a causal association supporting the presence of a fault in that class;
. the syndrome-class link represents, instead, a procedural relation suggesting the search in the class directly associated to the syndrome which reaches a sufficient degree of certainty.

Some faults share some symptoms and the same thing happens between symptoms and data. This suggests the need of propagating the values of data through the network and consequently reevaluating all probability values of symptoms and faults.

The search strategy we adopted (an ''opportunistic'' search) incorporates two main mechanisms:
- a backward mechanism leading from a fault hypothesis to data requests
- a forward mechanism implying the revalutation of all the hypoteses on the base of the collected data.
The resulting search strategy can be summarized in the following way:

1) **Inferential network firing**
   a. read the beam current value;
   b. it falls in the interval [5.14, 5.46] THEN continue the monitoting phase ELSE store in the investigation stack the variables representing syndromes and enter the following phase.

2) **Searching for the class containing the most probable solution**
   a. consider the syndrome at the top of the stack;
   b. obtain the datum necessary to the evaluation of the first unknown symptom contained in the syndrome;
   c. on the basis of the information gained from this datum revaluate the values of all the syndromes and reject those containing symptoms not according with the one suggested by the datum itself;
   d. IF the stack is empty THEN you cannot find a class containing a solution
   e. IF all the symptoms contained in the syndrome at the top of the stack are known and the syndrome value reaches a predefinite threshold THEN enter the following phase after having put in the stack the variables representing the faults contained in the class associated to that syndrome ELSE come back to step a.

3) **Solution refinement**
   a. consider the fault at the top of the stack;
   b. obtain the datum necessary to the evaluation of the first unknown symptom contained in the set of the effects of that fault;
   c. on the basis of this new information revaluate all the values of the fault contained in that class and reject those containing symptoms not according to the one suggested by the new datum;
   d. IF Tthe stack is empty THEN you cannot reach a precise diagnosis;
   e. IF the symptoms contained in the fault variables are all known and the value of the variable reaches a definite threshold THEN you reached a diagnosis ELSE come back to step a.

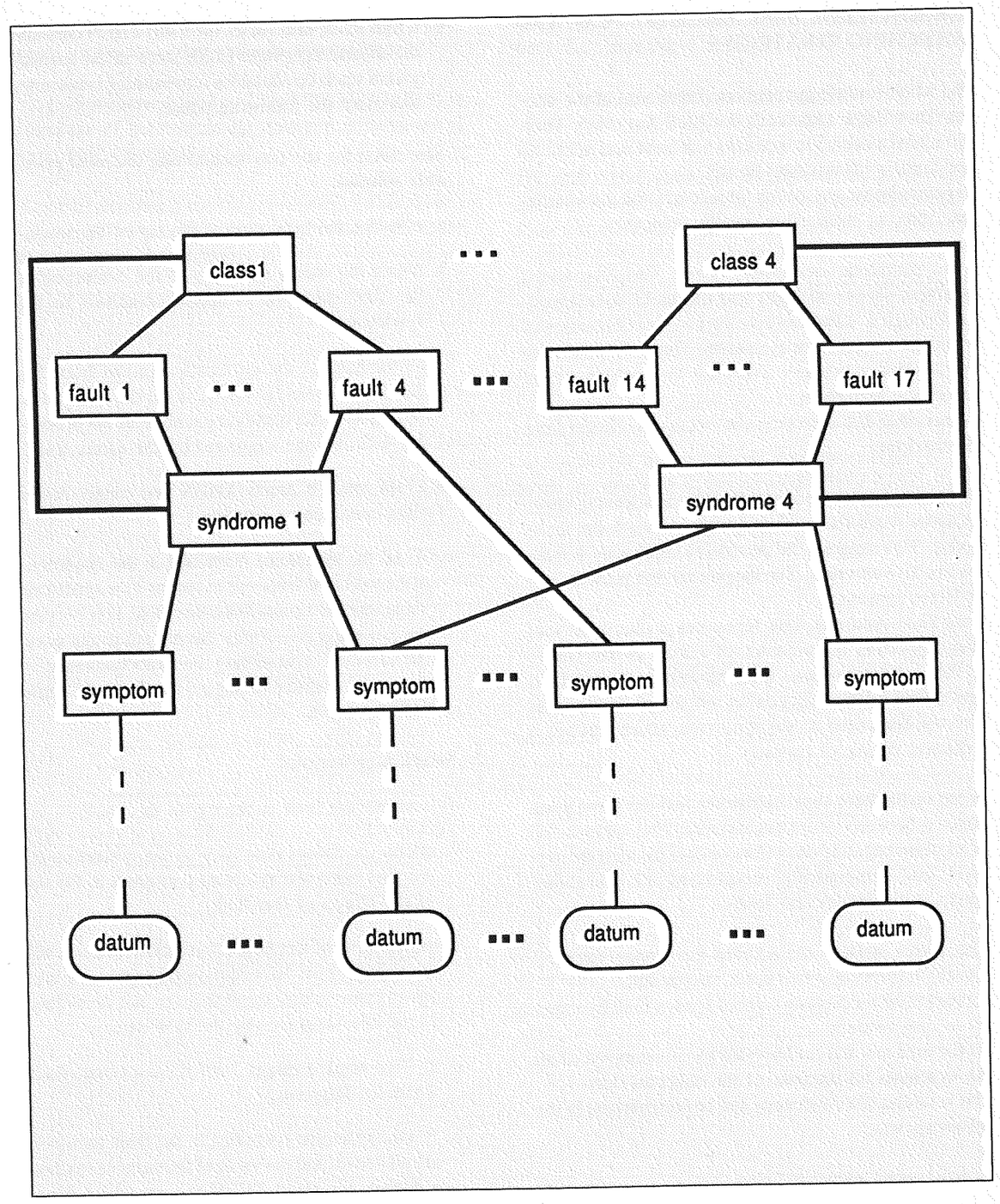

Fig. 6

6. CONCLUSIONS

The main results of this work consists in the definition of a general methodology to develop Expert Systems in plant mainteinance field, based on a triple knowledge model (componental, heuristic and decisional). In the future a the extension of the knowledge base is planned in order to cover the total number of faults. The next step is the integration on-line of SEDALwith the control system of the linear accelerator Clinac-4.

ACKNOWLEDGEMENTS

We wish to thank Prof.G.Degli Antoni for his indispensible suggestions in developing this work. We are indebted to the technical staff of our expert, from the Institute of ''Fisica Sanitaria - Ospedali Galliera'' (Genoa).

REFERENCES

[1] Cheesman, P., IN DEFENSE OF PROBABILITY, Proc. 9 th IJCAI, Los Angeles, 1985

[2] Clancey, W.J., HEURISTIC CLASSIFICATION, Artificial Intelligence Journal, n.27, 1985

[3] CLINAC 4, OPERATING MANUAL, Varian Ass., Palo Alto, 1972

[4] Genesereth, M.R., THE USE OF DESIGN DESCRIPTIONS in Automated Diagnosis, Artificial Intelligence Journal, n.24, 1984

[5] Hayes-Roth, F., D.A.Waterman, D.B.Lenat, (eds.) BUILDING EXPERT SYSTEMS, Addison-Wesley, Reading, 1983

[6] Hendee, W.R., MEDICAL RADIATION PHYSICS, Year Book Medical Pu., Chicago, 1970

[7] Kanal, L.N., J.F.Lemmer, (eds.), UNCERTAINTY HANDLING IN ARTIFICIAL INTELLIGENCE, Elsevier Science Pu., Amsterdam, 1986

[8] Karzmark, C.J., R.J.Morton, A PRIMER ON THEORY AND OPERATION OF LINEAR ACCELERATORS IN RADIATION THERAPY, Technical Report FDA 82-8181, U.S. Department of Healt and Human Services, Bureau of Radiological Healt, Dec., 1981

[9] Rossi, H., A.Wambersie, (eds.), ACCURANCY REQUIREMENTS AND QUALITY ASSURANCE OF EXTERNAL BEAM THERAPY WITH PHOTON AND ELECTRONS, Proc ICRU, 1986

[10] Spiegelhalter, D.J., A STATISTICAL VIEW OF UNCERTAINTY IN EXPERT SYSTEMS, in W.Gale (ed.), ''Artificial Intelligence and Statistics'', Addison-Wesley, Reading, 1985

[11] Spiegelhalter, D.J., PROBABILISTIC EXPERT SYSTEM IN MEDICINE: PROCTICAL ISSUES IN HANDLING UNCERTAINTY, Statistical Science, n.1, 1987

[12] The SAVOIR Expert System Package, PROGRAMMER GUIDE AND REFERENCE MANUAL, Intelligent Systems International Ltd., Redhill, 1987

[13] Weiss, S.M., C.A.Kulikowski, S.Amarel, A.Safir, A MODEL-BASED METHOD FOR COMPUTER-AIDED MEDICAL DECISION-MAKING, Artificial Intelligence Journal, n.11, 1978

[14] Zadeh, L.A., IS PROBABILITY THEORY SUFFICIENT FOR DEALING WITH UNCERTAINTY IN AI: A NEGATIVE VIEW, in [7]

[15] Duda, R., J.Gasching, P.Hart, MODEL DESIGN IN THE PROSPECTOR CONSULTANT SYSTEM FOR MINERAL EXPLORATION, in D.Michie (ed.) ''Experet Systems in the Micro-Electronic Age'', Edinburgh University Press, Edinburgh, 1979

WAVELET - BASED MATCHED FILTERING APPLIED TO NEUROMAGNETIC SIGNALS

G. Crosta, A. Paccanaro
Dipartimento di Scienze dell'Informazione
Università degli Studi di Milano

RIASSUNTO

Il rapporto segnale rumore nelle misurazioni di campi magnetici cerebrali evocati mediante stimolazioni sensoriali è molto basso. La normale procedura di estrazione del segnale fa uso di stimolazioni ripetute periodicamente e consiste in un'operazione di media delle misurazioni su un numero di cicli molto elevato (circa 500).
Le ''Wavelets'', una famiglia di funzioni ad un parametro di filtri adattati, sono state applicate con buoni risultati per estrarre il segnale evocato contenuto in un singolo ciclo. Inoltre, risultati sperimentali suggeriscono che, per misurazioni fatte in prossimita' del cranio in punti diversi, i segnali neuromagnetici evocati abbiano il medesimo valore tipico del parametro del filtro.

ABSTRACT

The signal to noise ratio in measurements of cerebral magnetic fields evoked by sensorial stimulation is very poor. The usual procedure for enhancing the neuromagnetic response signal applies to periodic stimuli and consists of averaging the detector output over a sufficiently large (about 500) number of cycles.
''Wavelets'', a one parameter family of matched filter functions, have been applied to enhance the neuromagnetic response signal contained in a single cycle with good results. Furthermore, experimental results suggest that neuromagnetic evoked signals are likely to have their own typical value of the filter parameter which is the the same for all measurements taken at different points near the skull.

1. INTRODUCTION

One major problem in neuromagnetics is the identification of electrical activity sources inside the CNS from non-invasive measurements of the magnetic field [1][2]. The solution procedure consists of three main operations on data:

i) acquisition by means of a given experimental setup;
ii) preprocessing, also known as ''filtering'';
iii) inversion i.e., application of a suitable model, which relates the desired, unknown quantities to the measured ones.

Very sensitive superconducting devices (SQUIDs) are used to measure the magnetic field induced by neural currents due both to spontaneous and evoked cerebral activity [3][4][5][6]. When studying evoked cerebral activity, experimental data are typically collected by applying a sensorial (visual, tactile, electrical...) stimulus to an individual and continuously measuring the magnetic field or the gradient thereof, in the vicinity of the head. It is usually expected that a distinctive feature, the ''signal'', show up in the recorded magnetic field in response to the stimulus.
Since the amplitude of the evoked signal is very low and since much electrical activity of unknown type goes on in the CNS at the same time, the resulting signal to noise ratio is very poor, despite shielding the experimental setup and the use of specially designed magnetic field gradiometers[3]. Enhancing the signal to noise ratio is therefore a mandatory task, which shall be carried out before any data inversion procedure.
The usual enhancement procedure applies to periodic stimuli and consists of averaging the detector output over a sufficiently large number of cycles s.t., the desired ''signal'' appears as a pulse out of the background.
The question arises whether some degree of enhancement could be obtained in real time i.e., on a single cycle, without any averaging.
One possible way is to look for a discontinuity or ''change'' in the detector output. The classical way of accomplishing this task is by matched filtering.
In the recent past ''wavelets'' have been proposed as one - parameter families of matched filters: they have been effectively applied to the detection of discontinuities in a variety of situations [7][8][9]. In the following we illustrate the design and application of wavelet matched filters to the preprocessing of some neuromagnetic signals.

1. Wavelet Filtering

Let $s \in L^2(\mathbf{R})$ denote a (time dependent), real valued and square integrable function; let $\| \cdot \|_2$ denote the L^2—norm and $\mathcal{F}[.]$ the Fourier transform operator. Consider the set of filter functions which are square integrable and even

$$\mathcal{G}_{ad} = \{ g \mid g \in L^2(\mathbf{R}),\ \mathrm{Im}\,[\mathcal{F}g] =_{\text{a.e.}} 0 \} \qquad (1.1)$$

Denote complex conjugation by *, convolution by $\circledast$, cross correlation by $\otimes$; introduce $\tilde{g}(t) :=_{\text{a.e.}} g^*(-t)$. Then the following can be easily shown

i) convolution of $\tilde{g}$ by s equals cross — correlation of g by s

$$(\tilde{g} \circledast s)(b) =_{\text{a.e.}} (g \otimes s)(b) \qquad (1.2)$$

i.e., explicitly

$$\int \tilde{g}(b-t)\, s(t)\, dt =_{\text{a.e.}} \int g^*(t-b)\, s(t)\, dt; \qquad (1.3)$$

ii) from the cross-correlation theorem
$(\mathcal{F}[\, g \otimes s\,] = [\mathcal{F}g]^* [\mathcal{F}s])$
$$(\tilde{g} \circledast s)(b) =_{\text{a.e.}} \int \exp[ib\omega]\, [\mathcal{F}g](\omega)\, [\mathcal{F}s](\omega)\, d\omega, \qquad (1.4)$$
where all integrals extend over $\mathbf{R}$.

A one-parameter family of wavelets can be defined by

$$g_a := \frac{1}{\sqrt{a}}\ g\left(\frac{t}{a}\right) \qquad 1.5$$

where a (>0) is the *scale parameter* and $g \in \mathcal{G}_{ad}$, hereinafter called the *analyzing wavelet*, satisfies $\| g \|_2 = 1$. From (1.5) it follows

$$g_a \in \mathcal{G}_{ad} \text{ and } \| g_a \|_2 = 1,\ \forall\ a > 0 \qquad (1.6)$$

Let $b \in \mathbf{R}$ and define the *wavelet transform* of s by

$$S(b, a) := \tilde{g}_a \circledast s \qquad (1.7)$$

As a consequence of (1.2), $S(.,.)$ satisfies

$$S(b, a) =_{\text{a.e.}} \frac{1}{\sqrt{a}} \int g^*\left(\frac{t-b}{a}\right) s(t)\, dt =_{\text{a.e.}} = \sqrt{a} \int \int \exp[ib\omega]\, [\mathcal{F}g](a\omega)\, [\mathcal{F}s](\omega)\, d\omega\ . \qquad (1.8)$$

In order to illustrate the role of the parameter a , the following may be considered. Assume supp $[\mathcal{F}g] \subseteq \{ \omega \mid 0 \le \omega_{\min} \le |\omega| \le \omega_{\max} \}$, where $\omega_{\min}$, $\omega_{\max}$ are given numbers. Then the last integral on the r.h.s of (1.8) extends over $\Omega_a := \{ \omega \mid 0 \le \frac{\omega_{\min}}{a} \le |\omega| \le \frac{\omega_{\max}}{a} \}$.

Convolution of the corresponding g_a by s provides then adjustable bandpass filtering: ''small'' values of a shall enhance fine scale details of s , whereas ''slow'' changes in s will be deemphasized. Very roughly speaking, the image of the plane $\{ a > 0, b \}$ through $S(.,.)$ describes $s(.)$ ''at every scale''.

A commonly used analyzing wavelet reads:

$$g(t) = \exp[\, i\omega_0 t] \exp\left[-\frac{t^2}{2}\right]. \qquad (1.9)$$

In addition to implementing bandpass filters, wavelets can serve as matched filters in a broad sense: this is the application considered herewith. It is well - known that the purpose of matched filtering is to determine how similar the process under test e.g., the s introduced above, is to a given feature, the ''signal'' $h \in L^2(\mathbf{R})$. According to the standard definition, the filter function $f (\in L^2(\mathbf{R}))$ matched to h is s.t., $[\mathcal{F}f]^* = [\mathcal{F}h]$. The filter function g_a, which can be shown to comply with condition 1.9, implemented in the following does *not* in general comply with the last condition: instead, the optimal parameters a has been determined by maximizing a quantity known as the *reciprocal noise figure*. More precisely, assume that supp $[s] = [0, T]$ and supp $[h] = \tau := [t_1, t_2] \subseteq T$. Since the desired feature, h, although buried in noise, is *a priori* known to occur in the interval $[t_1, t_2]$, then it makes sense to define the *input signal to noise ratio* ($\mathrm{SNR_I}$) by

$$\mathrm{SNR_I} := \frac{\int_\tau |s|^2\, dt}{\int_{\mathrm{supp}[s]/\tau} |s|^2\, dt} \qquad (1.10)$$

The corresponding quantity after filtering i.e., the *output signal to noise ratio* ($\mathrm{SNR_O}$), will then depend on a and read

$$\mathrm{SNR_O}(a) := \frac{\int_\tau |S(b, a)|^2\, db}{\int_{\mathrm{supp}[s]/\tau} |S(b, a)|^2\, db} \qquad (1.11)$$

The the *reciprocal noise figure* (RNF), which quantifies the enhancement yielded by filtering also depends on a and is defined by

$$\mathrm{RNF}(a) := \frac{\mathrm{SNR_O}(a)}{\mathrm{SNR_I}} \qquad (1.12)$$

In a practical implementation time is a discrete variable, therefore the discrete - time counterpart of the above definitions and properties shall apply. No new notation will however be introduced for quantities in the discrete setting.

2. PREPROCESSING EXPERIMENTAL DATA

A wavelet - based matched filter has been applied to enhance the neuromagnetic response signal contained in a single cycle. Experimental data have been kindly supplied by the Group at the *Istituto di Elettronica dello Stato Solido* in Rome (IT).

A subject has his wrist stimulated by a square wave current pulse; magnetic fields are detected by means of a nine - channel *dc SQUID*[10]. The stimulus lasts for 7ms and has a period $T = 300$ ms. The detector outputs are sampled at 1 kHz. The signal to be looked for is a time dependent component of the magnetic field, which is somatosensorially evoked through the stimulation of the median nerve.

2.1 Model calibration

In This section we shall refer to data obtained from a single channel (channel 2).
Averaging data over a sufficiently high number, N, of cycles typically yields the function $\langle s \rangle_N (t)$.
Fig. 1 shows the plot of $-\langle s \rangle_N (t)$, for $N = 350$, for t = 0..100 ms (plotting the opposite of $\langle s \rangle_N (t)$ improves understanding). The origin represents the instant of stimulation. The feature which a wavelet based matched filter shall detect out of single cycle data is represented by the arc lasting from $t_1 = 15$ ms to $t_2 = 21$ms and exhibiting a local maximum at 18ms.

The goal is achieved in two steps:
i) given the analyzing wavelet of (1.9), the family defined by (1.5), the average $\langle s \rangle_N (.)$, the values t_1 and t_2, replace s by $\langle s \rangle$ in 1.10 and 1.8, tune the filter to the signal; this amounts to finding the optimal value $\hat{a}$ which maximizes RNF(.) of (1.12);
ii) implement the filter on a single cycle (on a few cycles) and evaluate the results by computing the corresponding RNF(.).

The value $\omega_0 = 5.5$ (see 1.9) has been determined according to the heuristic procedure described by Kronland-Martinet [9].
All numerical computation have been carried out in extended precision (96 bit floating point) on a MacintoshIICX. Source codes have been written in G language under Lab View environment.
The plot of RNF *vs* a , determined according to step *i)* above, is shown in Fig. 2. The absolute maximum is seen to occur at $\hat{a} = .38$ s.t., $\mathrm{RNF}(\hat{a}) = 27.4072$. The optimal member of the analyzing wavelet family having been found, filtering of raw data can be implemented.
Figures 3 to 7, 9 and 10 which follow, show magnitudes of convolved signals i.e., |S(.,.)|.
Fig. 3 shows one of the best results. It should be noted that the electrical stimulus occurs at 243 ms, hence the response peak is expected at 261 ms. The filter output does indeed return said peak. Single cycle filtering however is not always capable of enhancing the expected peak that much: Fig. 4 pertains to a sequence of two cycles, which were filtered without averaging: stimuli of equal amplitude occur at 101 and 401 ms, respectively. The 419 ms peak is lower than the one occurring at 119 ms.
Of course filtering experimental data, which have been previously averaged over a number of cycles, consistently results in an increasing improvement. The corresponding plots are shown by Fig. 5 to 7. The signal averaged over 50 cycles is also shown for comparison (Fig.8).

2.2 Model validation

The wavelet based matched filtering has been applied to the detection of other features of the signal. In every case results were similar to those discussed above for the 18ms peak.

In order to filter out the features represented by the arc lasting from $t_1 = 35$ms to $t_2 = 53$ms and exhibiting a local maximum at 44ms (see Fig.1) another value of the parameter a had to be determined by the RNF procedure described in the previous section. The optimum value turned out to be $\hat{a} = 0.885$. Fig.9 shows the result of convolution with experimental data which have been previously averaged over 50 cycles.
A very interesting result has been found by extracting the same signal feature from other channels of the dc-SQUID, which were measuring the magnetic field at different points outside the skull at the same time. The optimum value of a for a particular feature turned out to be the same for all channels: e.g. for channel 7, as for channel 2, the values of a which best detect the 18ms and 44ms peak are respectively 0.38 and 0.885. Fig.10 shows the result of convolution with experimental data from channel 7 which have been previously averaged over 50 cycles; $a = 0.38$ to enhance the 18ms peak. These last results suggest that each feature of the neuromagnetic signal is likely to have its own typical value of the parameter a, which is the same for all measurements.
Of course, when data from other channels are being processed or other peaks are sought for, single cycle filtering does not always return the best results, as it has alredy been stressed in the previous sub-section: the input signal to noise ratio in these cases is the poorest by definition.

3. CONCLUSION

Preprocessing of raw neuromagnetic data by a wavelet based matched filter has been described. The only optimization step so far has been the determination of the scale parameter $\hat{a}$.
Different analyzing wavelets have not been considered in this preliminary work. The results suggest that the application domain be thoroughly explored, in search for improved performance. The ultimate goal is to produce, with the shortest possible delay, sets of filtered data which lend themselves to inversion by some adequate procedure.

ACKNOWLEDGEMENTS

The authors are particularly grateful to Prof. G. Degli Antoni for continuous support, discussions and guidance.
Furthermore they want to thank Prof. G. Haus who first suggested the use of wavelets for preprocessing raw neuromagnetic data.
Prof. G. L. Romani and Dr. V. Pizzella in particular and the whole Group at the *Istituto di Elettronica dello Stato Solido* in Rome (IT) are gratefully acknowledged for supplying digital recordings of raw experimental adata and for the useful comments of the results.

REFERENCES

[1] R. Harrop, H. Weinberg, P. Brickett, THE BIOMAGNETIC INVERSE PROBLEM : SOME THEORETICAL AND PRACTICAL CONSIDERATIONS, Phys.Med.Biol., Vol.32, N.12, pp. 1545-1557, 1987.

NOTE DI SOFTWARE 50

[2] B.N. Cuffin, THE ROLE OF MODELS AND COMPUTATIONAL EXPERIMENTS IN THE BIOMAGNETIC INVERSE PROBLEM, Phys.Med. Biol., Vol.32, N.1, pp. 33-42, 1987.

[3] G.L. Romani, S.J. Williamson, L. Kaufman, BIOMAGNETIC INSTRUMENTATION, Rev.Sci.Instrum., Vol.53, N.12, pp. 1815-1845, December 1982c.

[4] P. Carelli, V. Foglietti, I. Modena, G.L. Romani, MAGNETIC STUDY OF THE SPONTANEOUS BRAIN ACTIVITY OF NORMAL SUBJECTS, Il Nuovo Cimento, Vol.2D, N.2, pp. 538-546, March-April 1983.

[5] R. Hari, K.Reinikainen, E. Kaukoranta, M. Hamalainen, R. Ilmoniemi, SOMATOSESORY EVOKED CEREBRAL MAGNETIC FIELDS FROM SI AND SII IN MAN, Electroencephalography and clinical Neurophysiology, Vol.57, pp. 467-473, 1984.

[6] G.L. Romani, P. Rossini, NEUROMAGNETIC FUNCTIONAL LOCALIZATION : PRINCIPLES, STATE OF THE ART AND PERSPECTIVES, Brain Topography, Vol.1, N.1, pp. 5-21, 1988.

[7] J.M. Combes et al., WAVELETS - TIME FREQUENCY METHODS AND PHASE SPACE, Springer Verlag, Berlin, 1989.

[8] R. Kronland-Martinet, J. Morlet, A. Grossman ANALYSIS OF SOUND PATTERNS THROUGH WAVELET TRANSFORM, Int. J. of Pattern recognition and Artificial Intelligence, Vol.1, pp. 273-302, 1987.

[9] A. Grossman, M. Holschneider, R. Kronland-Martinet, J. Morlet, DETECTION OF ABRUPT CHANGES IN SOUND SIGNALS WITH THE HELP OF WAVELET TRANSFORMS, Advances in Electronics and Electron Physics, Suppl.19, Inverse problems, Academic Press, pp. 298-306, 1987.

[10] P. Carelli, RELIABLE LOW-NOISE DC-SQUIDS, Journal of Applied Physics, pp. 41-45, 1988.

ADDITIONAL REFERENCES

P. Dutilleux, A. Grossman, R. Kronland-Martinet APPLICATION OF THE WAVELET TRANSFORM TO THE ANALYSIS, TRANSFORMATION AND SYNTHESIS OF MUSICAL SOUNDS, Preprint A.E.S. 85th convention, Los Angeles, 1988.

A. Grossman, R. Kronland-Martinet, TIME AND SCALE REPRESENTATIONS OBTAINED THROUGH CONTINUOUS WAVELET TRANSFORMS, Signal Processing IV : Theories and applications, Elsevier Science Publ., pp. 475-482, 1988.

R. Kronland-Martinet, THE WAVELET TRANSFORM FOR ANALYSIS, SYNTHESIS AND PROCESSING OF SPEECH AND MUSIC SOUNDS, Computer Music Journal, Vol.12(4), pp. 11-20, 1988.

A. Paccanaro, TRATTAMENTO DI SEGNALI OTTENUTI MEDIANTE MISURAZIONE DI CAMPI MAGNETICI INDOTTI DA CORRENTI CEREBRALI, Tesi di Laurea in Scienze dell'Informazione, Univ. degli Studi di Milano, 1990.

S.J. Williamson, L. Kaufman, BIOMAGNETISM, J. of Magnetism and Magnetic Materials, Vol.22, pp. 129-201, 1981.

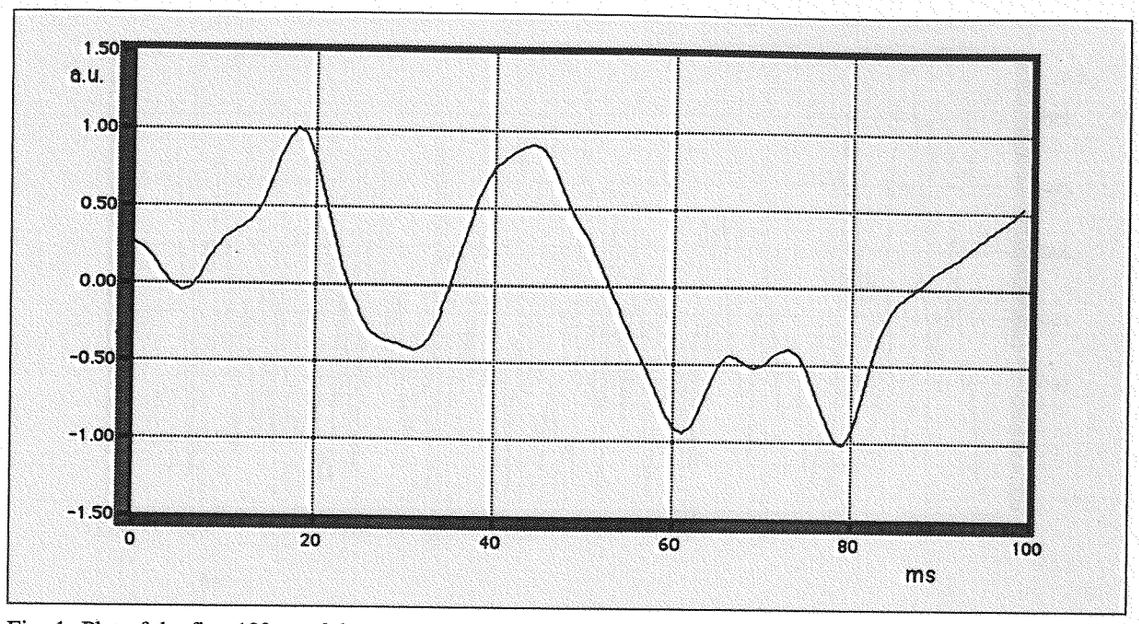

Fig. 1: Plot of the first 100ms of the opposite of $<s>_N(t)$ for N=350; data from channel 2. The origin represents the instant of stimulation.

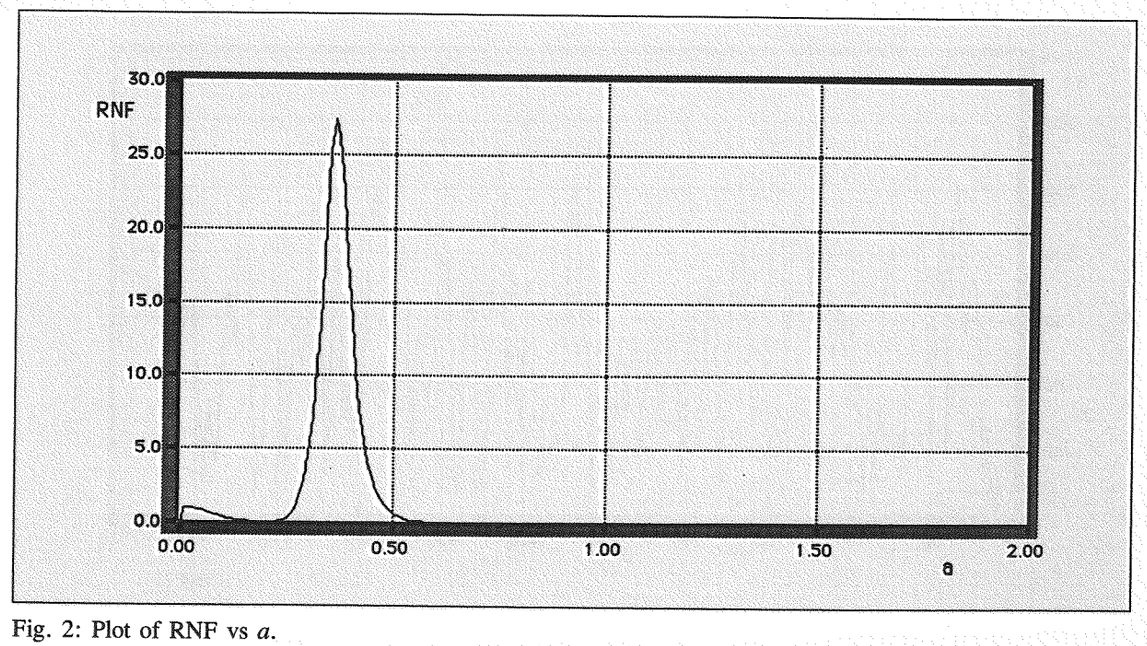

Fig. 2: Plot of RNF vs a.

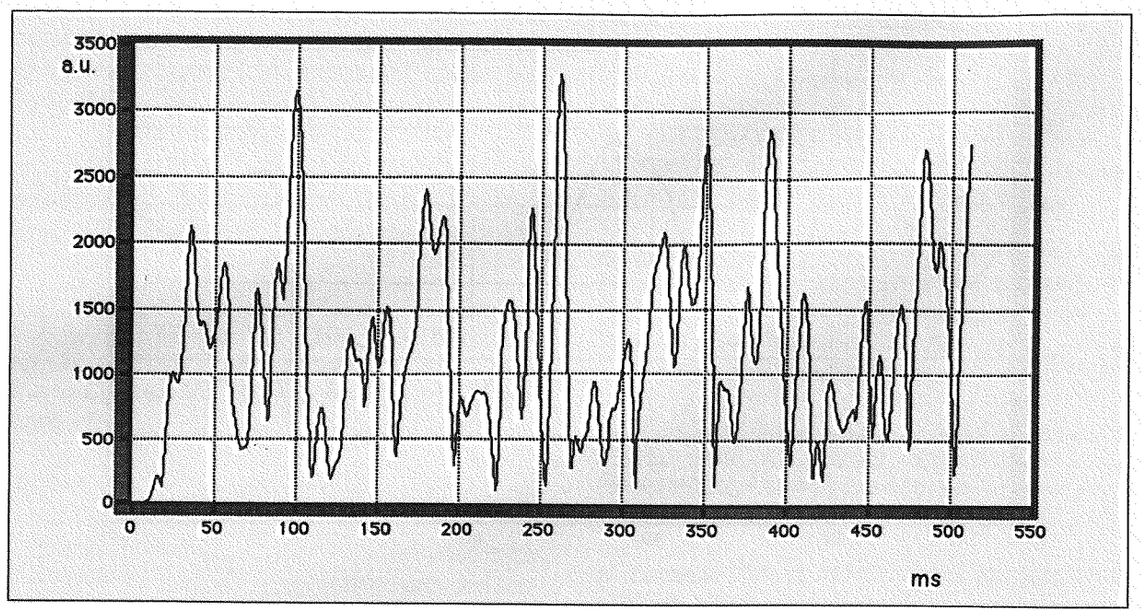

Fig. 3: Plot of the magnitude of the convolution between the analyzing wavelet with $a = 0.38$ and single cycle data. The stimulus occurs at 243ms. The response peak appears at 261ms.

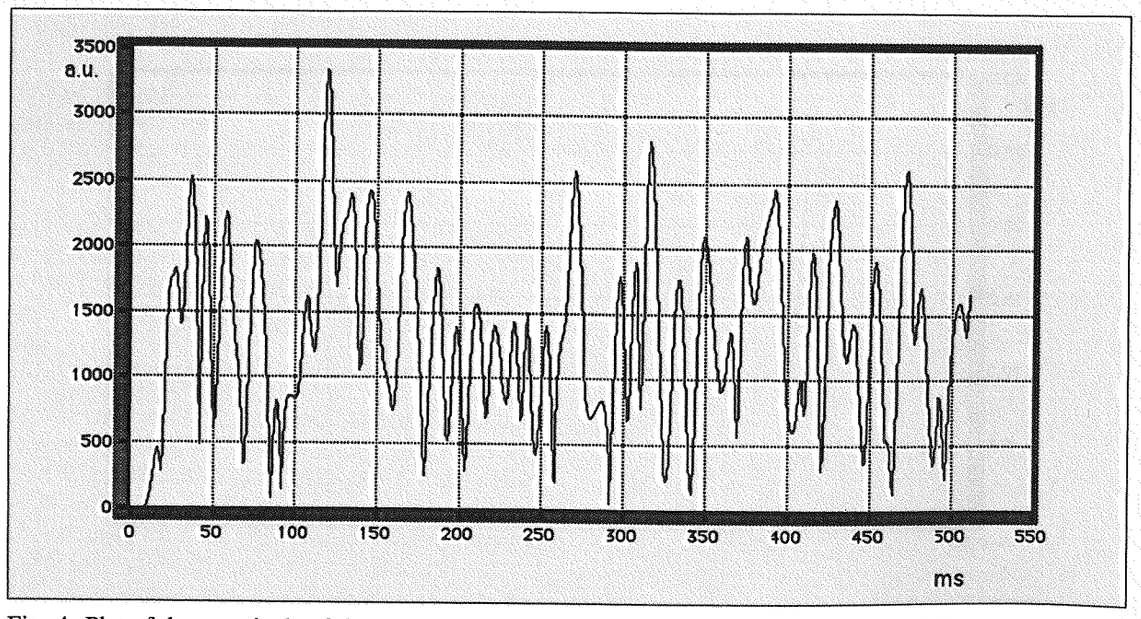

Fig. 4: Plot of the magnitude of the convolution between the analyzing wavelet with $a = 0.38$ and single cycle data. Stimuli of equal amplitude occur at 101 and 401ms. Response peaks appear at 119 and 419ms.

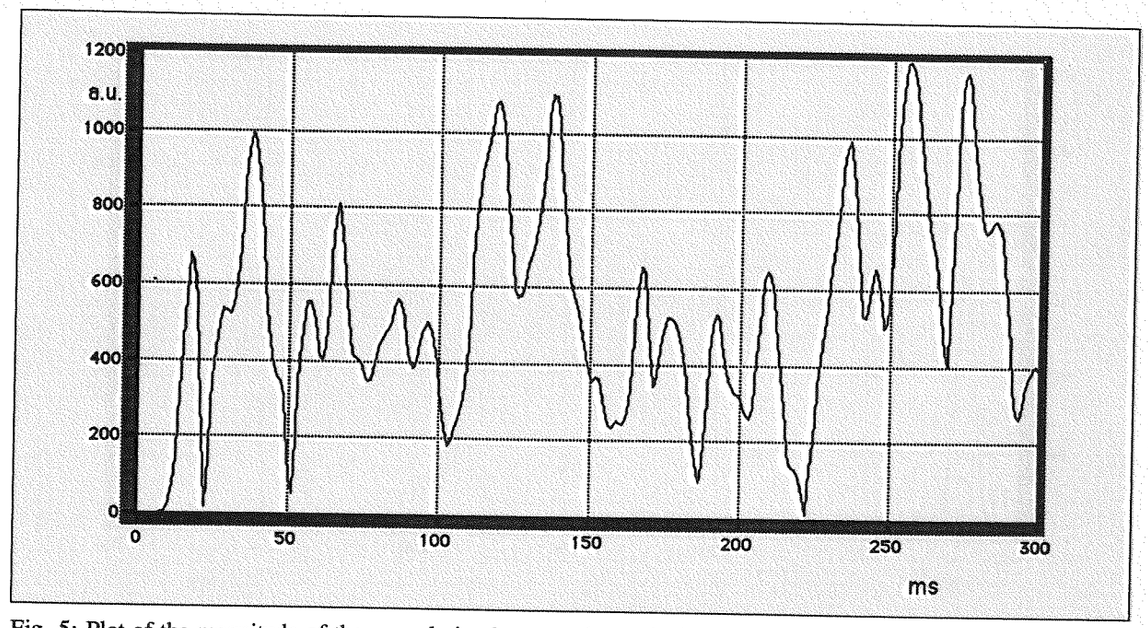

Fig. 5: Plot of the magnitude of the convolution between the analyzing wavelet with $a = 0.38$ and data averaged over 10 cycles. The origin represents the instant of stimulation.

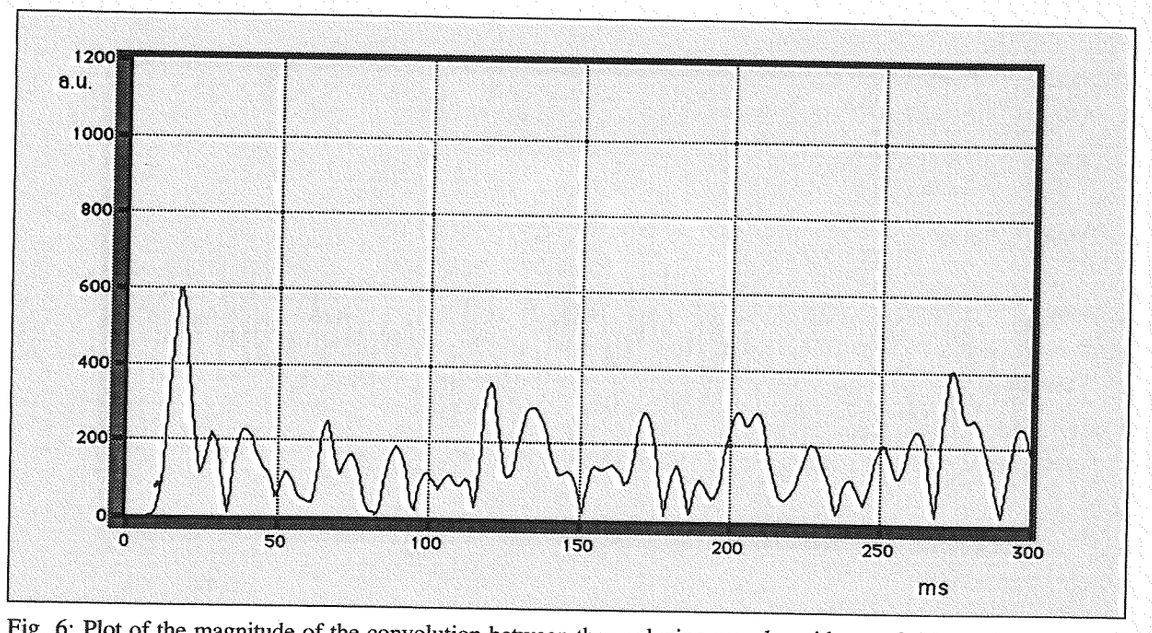

Fig. 6: Plot of the magnitude of the convolution between the analyzing wavelet with $a = 0.38$ and data averaged over 50 cycles. The origin represents the instant of stimulation.

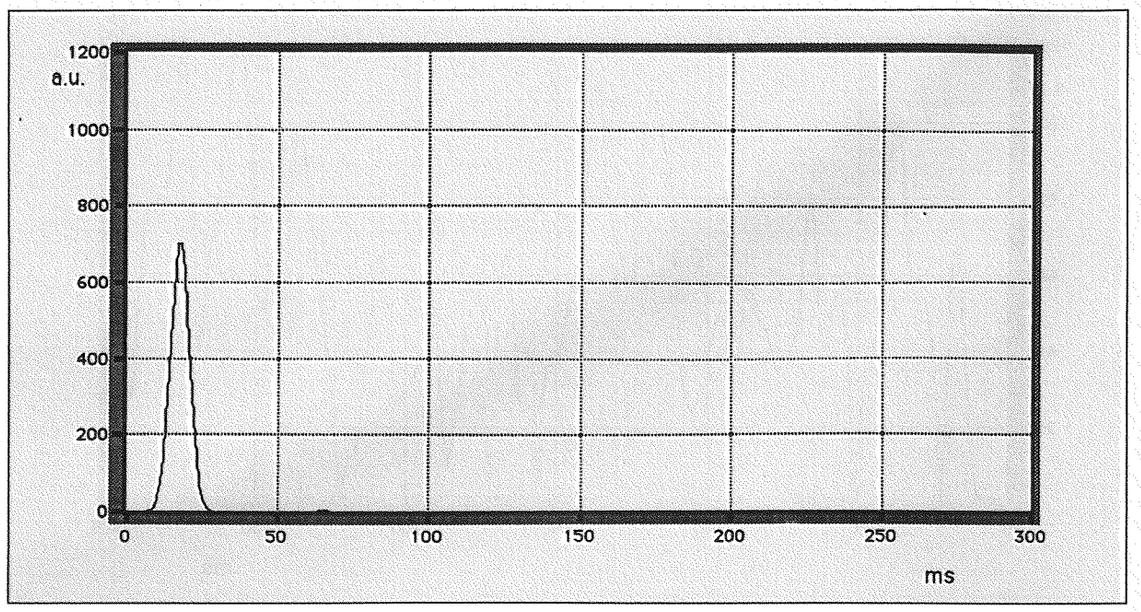

Fig. 7: Plot of the magnitude of the convolution between the analyzing wavelet with $a = 0.38$ and data averaged over 350 cycles. The origin represents the instant of stimulation.

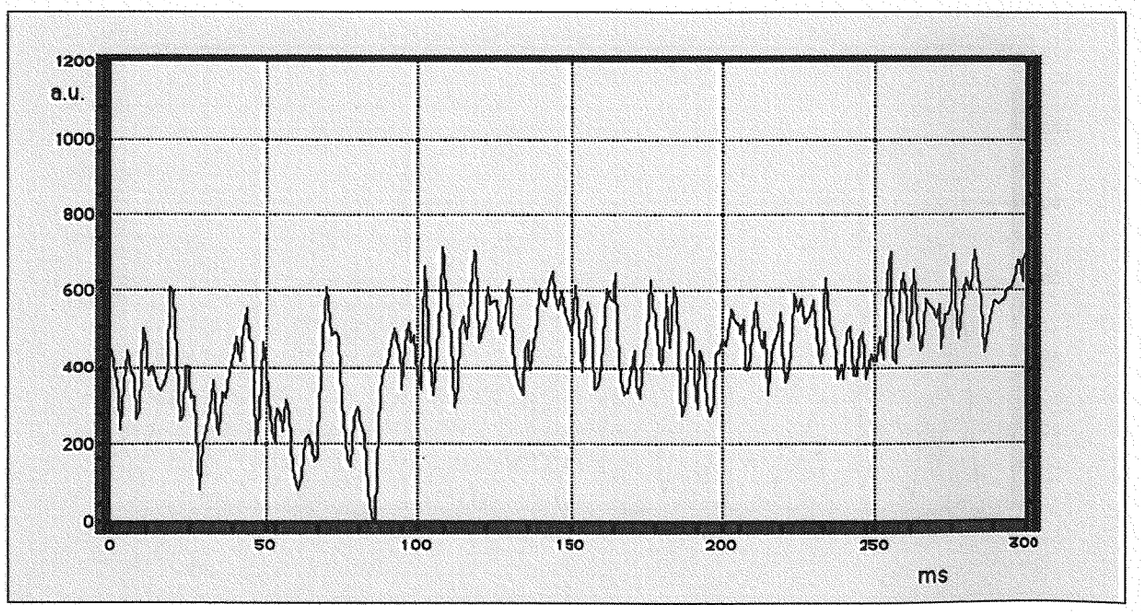

Fig. 8: Plot of data averaged over 50 cycles. The origin represents the instant of stimulation.

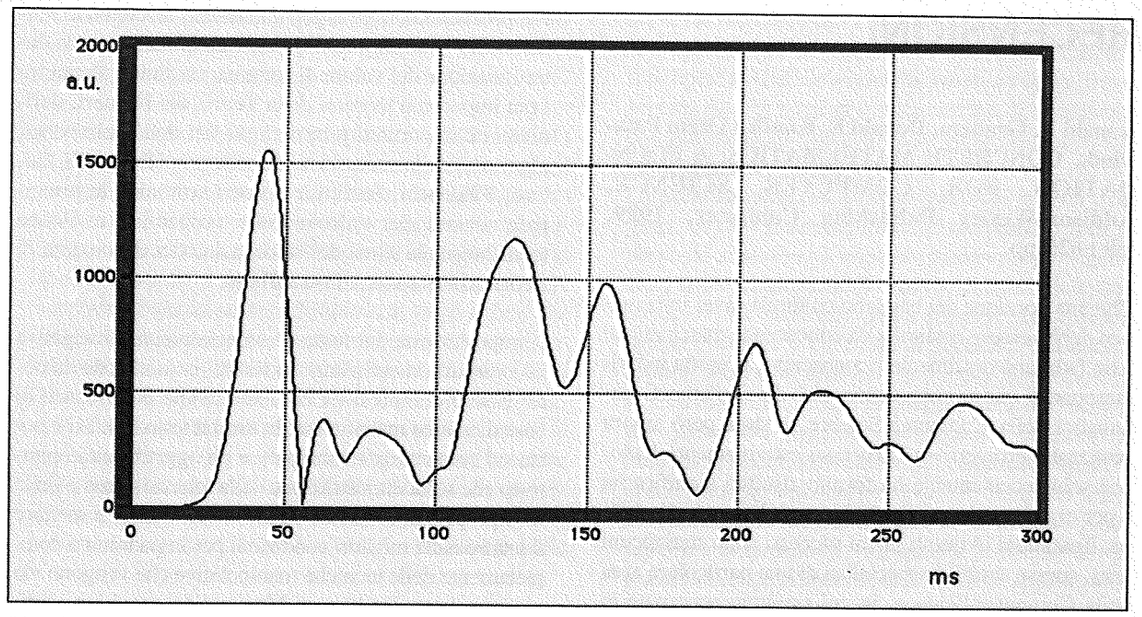

Fig. 9: Plot of the magnitude of the convolution between the analyzing wavelet with $a = 0.885$ and data from channel 2 averaged over 50 cycles. The origin represents the instant of stimulation.

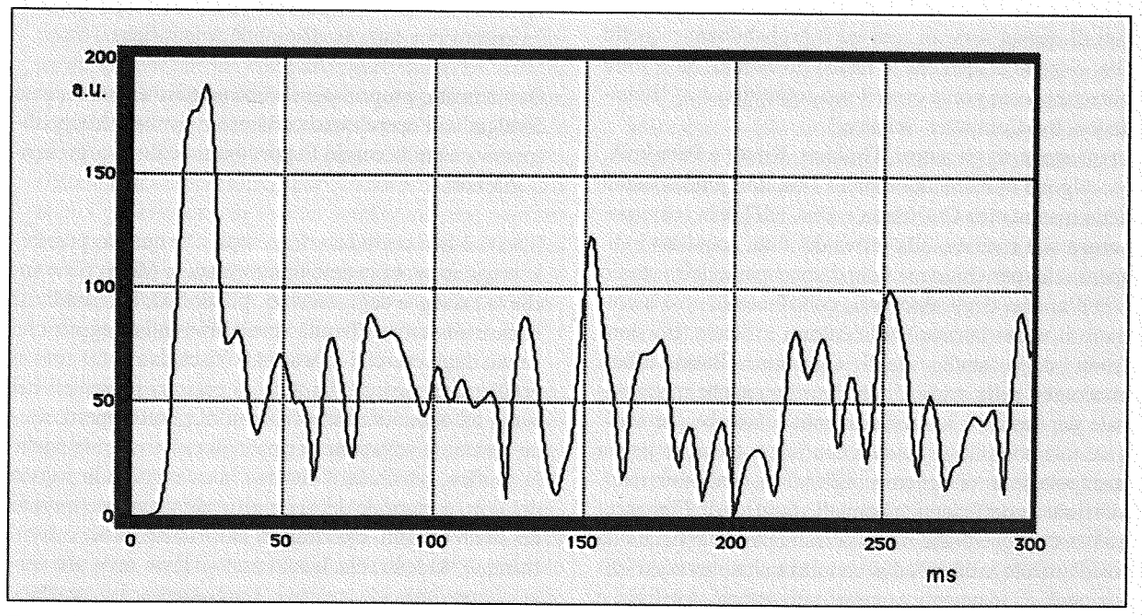

Fig. 10: Plot of the magnitude of the convolution between the analyzing wavelet with $a = 0.38$ and data from channel 7 averaged over 50 cycles. The origin represents the instant of stimulation.

RECENSIONI

Ronald L. Graham, Donald E. Knuth e Orem Patashnik, CONCRETE MATHEMATICS: A FOUNDATION FOR COMPUTER SCIENCE, Addison-Wesley Publishing Company, 1989, xiii+625 pp.

Che una porzione del bagaglio culturale di un informatico debba essere costituita da conoscenze matematiche è un fatto universalmente riconosciuto. L'esatta individuazione di queste conoscenze, cioè dei fondamenti matematici della Computer Science, è stata però sinora realizzata attraverso un processo estremamente lento, che solo recentemente ha fornito risultati significativi e per ora ristretti a particolari settori di questa disciplina. Testimoni di questi primi successi sono stati alcuni testi, spesso scritti da specialisti di una particolare area della Computer Science, in cui vengono accorpate in un quadro che vuole essere il più unitario possibile, nozioni spesso derivanti dai più disparati settori della Matematica.

''Concrete Mathematics'' costituisce uno tra gli ultimi nati di questa serie di testi ed è probabilmente quello che, rispetto ai testi che lo hanno preceduto, ha le mire più ambiziose; non a caso il suo sottotitolo è a ''Foundation for Computer Science''.
In questo testo gli autori, Graham, Knuth e Patashnik, raccolgono in maniera organica i capitoli fondamentali della matematica CONtinua e DisCRETA la cui conoscenza si è rivelata, alla prova dei fatti, requisito indispensabile per chiunque intenda occuparsi dello studio e dell'analisi degli algoritmi, contribuendo così a colmare il vuoto lasciato dalla cultura ufficiale, nei confronti dello studio degli algoritmi. Basta infatti cimentarsi nello studio di un algoritmo anche molto banale per rendersi immediatamente conto che per svolgere tale compito si devono conoscere tecniche per la manipolazione di oggetti combinatori quali sommatorie finite, coefficienti binomiali, numeri di Fibonacci ecc. ecc., oggetti che matematicamente nulla hanno da invidiare alle tecniche ed ai concetti trattati nei corsi istituzionali di matematica, come ad esempio Analisi I e Analisi II, ma che molto difficilmente vengono in essi anche solo accennati.

Più precisamente, gli argomenti trattati in Concrete Mathematics sono: tecniche per la manipolazione e la determinazione del valore di sommatorie finite, alcuni tra i più importanti teoremi della Teoria dei Numeri, definizioni e importanti proprietà, ai fini dell'analisi degli algoritmi, delle successioni dei numeri di Bernoulli, Eulero, Fibonacci, Stirling e numeri armonici, le principali nozioni del calcolo delle probabilità e alcune tecniche per la stima dei valori asintotici di funzioni di ricorrenza e sommatorie infinite.

L'impostazione del testo è essenzialmente divulgativa ed i vari argomenti vengono trattati in modo tale da poter essere compresi da chiunque abbia conoscenze di base di analisi matematica. In tutto il testo l'enfasi è posta sul come usare e manipolare gli oggetti trattati piuttosto che sulla dimostrazione della loro esistenza o delle loro proprietà, cercando in particolar modo di mettere il lettore nelle migliori condizioni per impadronirsi completamente delle tecniche manipolative che vengono via via introdotte. Un libro di Matematica quindi un pò diverso dal solito, in cui i dettagli tecnici prendono il sopravvento sull'essenza delle cose, ma, ribadiscono gli autori, pur sempre un libro di Matematica, che accanto al suo aspetto astratto non deve mai trascurare quello concreto se vuole continuare a rimanere un punto di riferimento per tutte le discipline scientifiche.

Ovviamente, proprio per il suo carattere estremamente divulgativo l'opera quasi certamente risulterà troppo dispersiva a chi vi cerchi l'approfondimento di argomenti già noti.

Il testo è ben corredato di esempi, che quando possibile vengono svolti in ambito informatico. Molto ricca anche la rassegna degli esercizi, più di 500, ben suddivisi in base alla loro difficoltà in sei differenti categorie che vanno dagli esercizi di warmup (riscaldamento) sino ai problemi di ricerca. Di tutti gli esercizi presentati nel testo, ad eccezione dei problemi di ricerca aperti, viene fornita la soluzione in appendice. Ben curata anche la grafica, particolare curioso in quasi tutte le pagine del libro a margine del testo ufficiale vengono riportati dei Math Graffiti, cioè frasi di illustri matematici o commenti di studenti, che hanno usato il testo nelle sue versioni preliminari, relativi l'argomento ivi trattato. Alcune di queste frasi sono molto utili per una maggiore comprensione del testo, altre sono semplici battute;

nel complesso comunque questi graffiti contribuiscono a mettere il lettore più a proprio agio nei confronti del testo.

In conclusione, Concrete Mathematics è un buon testo di Matematica scritto da informatici soprattutto per gli informatici, ben curato in tutti i suoi dettagli ed estremamente indicato per la fase di formazione di un qualunque informatico e di chiunque gli algoritmi non si accontenta solo di subirli ma li vuole anche capire. A supporto di questa nostra affermazione valga il fatto che un corso di Concrete Mathematics è presente nel piano di studi del corso di laurea in Computer Science della Stanford University sin dal 1970. Viene allora spontaneo chiedersi: come mai non si è ancora realizzata la necessità di inserire un corso con questo taglio anche nel piano di studi dei corsi di Informatica delle università italiane?

Danilo Bruschi

Branko Soucek, NEURAL AND CONCURRENT REAL-TIME SYSTEMS, Wiley, New york, 1989, pp. XVIII-387, L. 31.65.

Secondo le intenzioni dell'autore, il volume viene proposto come un libro di testo per i sistemi di elaborazione di VI generazione con particolare interesse verso le applicazioni real-time. La messa a punto di elaboratori a parallelismo massiccio e lo sviluppo di algoritmi in grado di esibire capacità di apprendimento ed autoadattamento rappresenta, per l'autore, la base di una nuova generazione di sistemi intelligenti che operano in tempo reale. Le applicazioni tipiche di questi sistemi sono l'acquisizione di dati, l'elaborazione di segnali, la visione, il controllo di processi, la robotica, i sistemi di supporto alle decisioni, i simulatori evoluti, il CIM etc.
Il volume è diviso in due parti. Nella prima, intitolata processi intelligenti, viene presentata una rassegna di algoritmi con esempi di applicazioni real-time nei settori precedentemente indicati. I primi due capitoli fungono da introduzione tecnica all'elaborazione di segnali. Altri due capitoli sono dedicati agli algoritmi neuronali e alle loro applicazioni. Il Capitolo 4 è di particolare interesse in quanto fornisce una vasta panoramica sulle applicazioni del neuronale a problemi real-world, argomento quasi mai trattato in modo ampio e aggiornato. L'ultimo capitolo di questa prima parte è dedicato alle applicazioni in tempo reale dei metodi classici di Intelligenza Artificiale: sistemi basati a regole, frames, schemi etc.; il tema dell'integrazione di queste tecniche con quelle di tipo neuronale viene purtroppo appena accennato.
La seconda parte del volume tratta dei sistemi intelligenti, ovvero le architetture che dovrebbero fornire il supporto adatto per l'implementazione degli algoritmi citati nella prima parte. Il capitolo di apertura è nuovamente un'introduzione, questa volta all'elaborazione in tempo reale per il controllo di processi. Ciascuno dei capitoli successivi analizza invece una particolare classe di architetture. Anche in questo caso, come nella prima parte del volume, il capitolo dedicato alle implementazioni neuronali VLSI costituisce un motivo di interesse per chi desidera avere lo stato dell'arte sui temi tecnologici scottanti legati al neuronale. Ben tre capitoli si occupano di transputer a diversi livelli: programmazione concorrente in OCCAM, applicazioni in tempo reale di sistemi basati su transputer, disegno di architetture basate su transputer (p.es. MEGAFRAME).
Il volume si chiude con una serie di previsioni riguardanti le tendenze del mercato in vari settori legati al tema delle tecnologie hardware e software per sistemi intelligenti.
Uno degli aspetti più interessanti che gli elaboratori di VI generazione sembrano voler offrire è l'integrazione fra i due maggiori approcci per il trattamento della conoscenza: quello simbolico e quello neuronale o subsimbolico. Il riconoscimento della complementarietà dei due metodi e la capacità di implementarli in modo integrato ed efficiente sono i dati distintivi di questa nuova generazione. Il volume di Soucek, a questo riguardo, presenta in modo dettagliato i pezzi costitutivi dei sistemi real-time di VI generazione senza però fornire idee circa il modo di assemblarli. Questo genera nel lettore una certa senzazione di eterogeneità e disorganicità nella materia trattata che, peraltro, riunisce un insieme davvero enorme di concetti, approcci e tecnologie. In tal senso, credo che il merito maggiore dell'opera sia essenzialmente quello di fornire lo stato dell'arte riguardo ad un insieme di tecnologie estremamente attuali, fornendo al lettore un insieme di informazioni utili per comprendere le tendenze e le direzioni più promettenti sui temi di ricerca applicata.

Nicolò Cesa-Bianchi

Samuel J. Leffler, Marshall Kirk McKusick, Michael J. Karels, John S. Quarterman, THE DESIGN AND IMPLEMENTATION OF THE 4.3BSD UNIX OPERATING SYSTEM, Addison-Wesley, 1989, pp 471

In questi ultimi anni, l'esigenza sempre pressante da parte degli utenti di una standardizzazione del software e delle architetture dei sistemi, da contrapporsi ai cosidetti sistemi *proprietary*, ha prodotto un cambiamneto radicale nel modo di concepire il mercato da parte dei costruttori, determinando la nascita della strategia denominata *sistemi aperti* (open system). La strategia si propone di diffondere, su macchine di produttori diversi, alcune tecnologie hardware e software comuni, scelte in base al consenso che esse riscuotono presso gli utenti.
Le ultime vicende del mercato informatico, determinate dalla formazione di OSF (Open Software Foundation) e UNIX International, non fanno altro che rafforzare questa tendenza. Infatti, i due organismi, a cui fanno parte i maggiori costruttori mondiali, intendono fornire, pur con metodi diversi, un ambiente operativo totalmente standard. L'indiscusso candidato per questo processo di standardizzazione, considerato da entrambi gli organismi come l'elemento di riferiemnto per i propri lavori, è il sistema UNIX.
La storia di UNIX è nota a tutti. È noto anche il fatto che a partire dal primo ''prototipo'' progettato da Thompson nel lontano 1969, UNIX ha subito una complicata genesi, proliferandosi in diverse versioni.
Si distinguono però in questa intrecciata situazione due principali scuole, campeggiate rispettivamente dall'AT&T e dall'Università di Berkeley, che hanno dato vita a due distinte versioni di UNIX: System V e BSD (Berkeley Software Distribution).
Per un utente finale che osserva UNIX dal punto di vista funzionale probabilmente le due versioni non si discostano di molto fra di loro. Ma la loro profonda differenza architetturale ha un peso significativo sulla progettazione delle applicazioni e sul lavoro di porting di UNIX su diverse macchine. Capire questa differenza, e di conseguenza la struttura interna delle versioni System V e BSD, è ormai un imperativo per progettisti e programmatori in ambienti UNIX.
Sfortunatamente, la letteratura esistente non aiuta molto il lettore in questo processo di apprendimento. Mentre le librerie abbondano di libri che presentano gli aspetti più esterni di UNIX, come i comandi, i tool e le librerie per la programmazione, poco è stato fatto per svelare i suoi aspetti più nascosti, cioè l'architettura del kernel.
Fanno eccezione a questa affermazione due ottimi libri recentemente disponibili in librerie. Il primo, di Maurice J. Bach (*The Design of the Unix Operating System*, Prentice-Hall, 1986), riguarda principalmente l'architettura interna del System V. Il secondo, dal titolo *The Design and Implementation of the 4.3BSD UNIX Operating System*, soggetto di questa recensione, è dello stesso argomento ma si riferisce all'ultima versione UNIX di Berkeley, 4.3BSD, rilasciata nel 1986.
Il libro, realizzato dai più qualificati progettisti di UNIX, ad incominciare da McKusick, l'autore di un file system che porta il suo nome, descrive in dettaglio gli aspetti architettuali ed implementativi del kernel 4.3BSD. La descrizione fa luce sulle strutture dati e sugli algoritmi utilizzati per il disegno e l'implementazione di tutte le componenti del sistema operativo, a partire dal livello delle system call. Gli autori si soffermano anche sul socket, un particolare meccanismo di IPC di Berkeley, e sull'implementazione dei noti protocolli di comunicazione TCP/IP.
Il libro è suddiviso in cinque parti. La prima, anche se è chiamata ''overview'' dagli autori, in realtà presenta in modo sufficientemente dettagliato gli elementi costituenti i servizi del kernel, come gli interrupt, le system call e le operazioni di accounting. In questa parte del libro, veniamo a conoscenza anche dei futuri progetti dell'Università di Berkeley per migliorare la propria versione, perchè, come sostengono gli autori, ''4.3BSD is not perfect''.
La seconda parte riguarda i processi ed inizia con il trattamento dei meccanismi per la loro gestione (scheduling, context switching...). Prosegue con l'analisi dettagliata della memoria virtuale, insieme alla storia della relativa evoluzione. L'architettura hardware di riferimento è quella di un VAX. Tutte le principali strutture dati e gli algoritmi per la gestione delle pagine di memoria sono riportati in dettaglio. La seconda parte si conclude quindi con il trattamento delle system call per la creazione e terminazione dei processi e la descrizione del meccanismo di swapping.
Il sistema I/O costituisce l'argomento della terza parte del libro, in cui l'implementazione del file system è descritta in dettaglio. Gli autori si soffermano sulla nuova organizzazione del file system 4.3 BSD e sugli elementi innovativi come il cylinder group, la lunghezza variabile dei nomi di file, il link simbolico. La co-

struzione dei device driver e la gestione dei terminali completano gli argomenti di questa parte. È interessante segnalare l'esempio dettagliato di un driver per i dischi.
La quarta parte è dedicata alla comunicazione fra i processi e, in generale, ai protocolli di rete. In essa troviamo innanzitutto un trattamento dettagliato di socket e dei meccanismi di comunicazione in rete, quindi una descrizione dei protocolli di rete, con particolare riferimento a TCP/IP.
Il libro si conclude con un argomento importante ma spesso trascurato: l'inizializzazione del sistema. I dettagli si riferiscono alle problematiche di inizializzazione e di configurazione del kernel ed all'importanza dei file messi a disposizione all'utente per questo lavoro.
Il libro è corredato di disegni, di tabelle riassuntive, di esercizi (alla fine di ogni capitolo) a tre diversi livelli di difficoltà e, infine, di riassunti degli algoritmi sotto forma di pseudo-C.
Oltre alla chiarezza nella presentazione degli argomenti, il libro ha il pregio di voler giustificare le affermazioni degli autori con frequenti e complete tabelle che riportano dati statistici e sperimentali, come nel caso del capitolo relativo al file system. Inoltre è prezioso il glossario alla fine del libro che fornisce precisi chiarimenti sui termini a cui spesso associamo una definizione molto approssimata.
Un neo (molto piccolo) possono essere la presentazione non proprio ordinata delle system call e lo scarso numero degli schemi riassuntivi degli algoritmi che, in alcuni casi, aiuterebbero meglio il lettore a riordinare le proprie idee.
Il libro, proprio per il livello di dettaglio con cui tratta gli argomenti, è comprensibile solo per chi possiede una discreta conoscenza relativa a tutti gli aspetti di un sistema di calcolo, soprattutto agli elementi di un sistema operativo. Personalmente, al contrario di quanto sostengono gli autori nella prefazione, penso che i semplici lettori ''curiosi'' incontreranno non poche difficoltà. Invece, per i progettisti di sistemi operativi e programmatori in ambiente UNIX costituisce senz'altro un libro di riferimento che deve essere consultato e capito a fondo.

Le van Huu

AVVENIMENTI

SISTEMI ESPERTI PER LA DIAGNOSTICA E LA CONFIGURAZIONE CORSO FAST 28 SETTEMBRE 1990

Carlo Ferigato
Dipartimento di Scienze dell'Informazione
Università degli Studi di Milano

Nella giornata di venerdì 28 settembre 1990 si è svolto, presso il palazzo della Federazione delle Associazioni Scientifiche e Tecniche (FAST), a Milano, il seminario dedicato ai sistemi esperti per la diagnostica e la configurazione. La giornata di studio si è inserita nel corso, dal titolo ''Tecnologie del Software'', organizzato da FOIST (Fondazione per lo Sviluppo e la Diffusione dell'Istruzione e della Cultura Scientifica e Tecnica) in collaborazione con la società Inferentia e l'Università degli Studi di Milano.

Per introdurre i partecipanti al problema il primo intervento (Alberto Pozzi per Inferentia) è stato dedicato ad una presentazione delle tecniche di rappresentazione della conoscenza utilizzate in Intelligenza Artificiale con particolare riferimento alla ''classificazione'' ed all'integrazione tra ''deep'' e ''shallow'' knowledge.

Ha proseguito Claude Vogel che, dopo una chiara esposizione del significato dell'Intelligenza Artificiale come disciplina e del ruolo che in questa hanno i sistemi esperti, ha presentato la metodologia di acquisizione della conoscenza da lui sviluppata per il CISI: KOD (Knowledge Oriented Design) che è attualmente in avanzata fase di automatizzazione e verrà distribuita entro la fine del 1990 su varie piattaforme hardware. Utilizzata sino ad ora ''a mano'' in decine di applicazioni per Intelligenza Artificiale realizzate dal CISI in Francia, guida l'ingegnere della conoscenza, o chi interroga l'esperto, a pensare in modo rigoroso alle manifestazioni verbali degli oggetti del dominio, alle proprietà che li definiscono, alla loro localizzazione spaziale; alle manifestazioni verbali delle azioni che sugli oggetti possono venire eseguite ed alle relazioni di dipendenza causale le legano ed infine alle manifestazioni verbali delle regole e dei vincoli che sul dominio da rappresentare sono definiti. Il materiale raccolto durante vari colloqui (guidati e non) con l'esperto viene utilizzato al fine di classificare oggetti ed azioni in tassonomie e di utilizzare appropriatamente vincoli e regole (schemi). La metodologia ha dimostrato (sul campo) che tre mesi sono sufficienti (indipendentemente dal dominio) per concludere la costruzione della base di conoscenze.

La giornata è proseguita con la presentazione di alcune applicazioni: Maurizio Dosso ha presentato il sistema di diagnosi sviluppato da O-Dati per il servizio assistenza tecnica di Olivetti. È basato sulla shell verticale TestBench (Carnegie Group) che vuole essere un compromesso tra un approccio ''a regole'' ed un approccio ''model based'' al problema della diagnosi. TestBench consente la costruzione di una tassonomia di concetti (guasti possibili) organizzati gerarchicamente attraverso relazioni ''caused-by'' mentre, associate ai nodi, sono

famiglie di test (per individuare la causa), istruzioni per la riparazione e regole per modificare dinamicamente la struttura della base di conoscenze: possono venire sostituiti valori all'interno degli oggetti classificati, riordinati e/o temporaneamente rimossi gli oggetti nella tassonomia (questo influisce sull'ordine della ricerca del guasto (che è depth- first)). Inoltre, una volta realizzato, il sistema è facilmente distribuibile su PC, permettendo così una diffusione capillare alla rete di assistenza. Il sistema si dimostra interessante in quanto persegue esplicitamente lo scopo di diminuire i costi di manutenzione col diminuire degli errori di diagnosi, delle scorte di ricambi e dei tempi in cui una macchina da riparare deve restare ferma, ed implica inoltre una riorganizzazione del lavoro utilizzando tecnici con minori competenze, centralizzando le decisioni sulle metodologie di diagnosi (e sulla produzione di documentazione relativa) e controllando in modo più diretto gli interventi di manutenzione delle consociate.

Anche Inferentia ha presentato un prototipo di sistema esperto nella configurazione: PIE (Progetto Isole Energetiche) il cui compito è quello di dimensionare impianti di co-generazione (calore ed elettricità) su scala locale, rispettando le esigenze dell'utenza, i vincoli ambientali, utilizzando le componenti appropriate (pompe di calore, motori Diesel, caldaie Turbogas...) garantendo un ritorno economico adeguato. Implementato su PC, realizza un'integrazione tra MS-Windows ed HP-New Wave (per un'interfaccia utente di buona qualità), il runtime di Nexpert Object (con il quale è implementato il nucleo del sistema esperto), il linguaggio DB3 (per la realizzazione della Base di dati dalla quale il sistema attinge informazioni) e routines Fortran, linguaggio con il quale è stato costruito un simulatore del comportamento dell''isola energetica'' (che utilizza tra l'altro l'algoritmo del simplesso). Alberto Pozzi ne ha approfittato per parlare di problemi di integrazione fra tecniche di Intelligenza Artificiale e Ricerca Operativa.

Hanno concluso la giornata Clara Bagnasco e Paola Petrin (per la società Quinary). Il loro intervento è stato incentrato sulla risoluzione di un problema di configurazione utilizzando sistemi ibridi di rappresentazione della conoscenza. In particolare hanno analizzato il ruolo giocato dalle ''gerarchie di specializzazione'' e dagli algoritmi di classificazione che su queste possono operare con strumenti di guida alle inferenze che il sistema compie, esplicitando (uniche a far ciò) il tema della semantica dei programmi realizzati con tecniche di Intelligenza Artificiale. Hanno infine presentato un esempio di configurazione basato sul sistema di rappresentazione ''Querelle'' (nel quale l'algoritmo di classificazione è implementato) e sulla shell QSL (sviluppati da Quinary stessa).

EDITO DA CALEIDOGRAF S.R.L.
VIA L. DA VINCI N.4/6 TEL. 039 - 587541
22058 OSNAGO (CO)

Spedizione in abbonamento postale, Gruppo IV, 1° semestre.
Pubblicità Inferiore al 70%
N. 50, Dicembre 1990

TAXE PERCUE
TASSA RISCOSSA
P.P. P.T. Piacenza F