UNIVERSITÀ DEGLI STUDI DI FIRENZE

DIPARTIMENTO DI INGEGNERIA ELETTRONICA

Rapporto interno n. 900401

Studio di un'architettura di calcolo parallelo basata su Transputer.

Dott.Ing. Lorenzo Capineri

Laboratorio Ultrasuoni e Controlli Non Distruttivi

50139 FIRENZE (ITALY) – Via S.Marta, 3 – Tel.47961 Telex 572460 Unifi

INDICE

Introduzione.

Le tecniche di ricostruzione di immagini basate sull'impiego degli ultrasuoni hanno un interesse crescente in campi applicativi come la biomedica ed i controlli non distruttivi. In generale si possono ottenere immagini che rappresentano strutture interne al mezzo investigato (ricostruzioni 3-D) o distribuzioni bidimensionali di parametri acustici (ricostruzioni 2-D o tomografie). Nel caso specifico della biomedica assume particolare importanza la ricostruzione 3-D dinamica, ovvero l'analisi del movimento e dei cambiamenti di forma nel tempo dell'organo investigato; ad esempio in cardiologia certe anomalie nei movimenti del cuore sono indicativi di patologie.

Queste tecniche si sono sviluppate in conseguenza ai progressi ottenuti nell'elettronica di front-end (sonde ad ultrasuoni e cortine ad alte prestazioni, metodi di focalizzazione, amplificatori integrati a guadagno variabile, convertitori A/D) e alla disponibilità di mezzi di calcolo sempre più potenti ad un costo relativamente basso. In generale tutte le tecniche di ricostruzione richiedono l'elaborazione di una grande massa di dati e attualmente i risultati non sono mai ottenuti in tempo reale [7]. In campo medico la caratteristica del tempo reale permetterebbe di seguire gli eventi mentre avvengono e quindi si potrebbero correlare con altri dati fisiologici disponibili con sistemi di monitoraggio standard.

Quindi la diffusione di queste tecniche rimane essenzialmente legata all'impiego dei supercomputers che con le loro elevate prestazioni riescono a fornire risultati in tempo quasi reale. Il fattore tempo di calcolo risulta fondamentale anche nel campo dei controlli non distruttivi, in quanto non è concepibile un apparato per il controllo in serie di pezzi meccanici la cui risposta si misura in termini di ore di tempo di calcolo su un Personal Computer oppure una risposta in termini di minuti con un

improponibile collegamento in rete con un centro di calcolo.

Fortunatamente in molte applicazioni l'algoritmo impiegato si può parallelizzare in modo da accelerare la sua esecuzione pensando ad una implementazione su un'architettura di calcolo parallelo.

In questo senso sono svolte ricerche per rendere paralleli algoritmi che richiedono l'impiego di supercomputers, ad esempio quelli impiegati nella SAR, SAFT, tomografia 2-D, ricostruzione 3-D, ray-tracing.

Un aspetto interessante sarà quello di capire lo scenario attuale nel campo dei calcolatori ad alte prestazioni e le tendenze della ricerca e della produzione nel settore dei sistemi di calcolo parallelo.

In questo campo sono d'interesse le reti di calcolo distribuito che impiegano i " Transputer ". Questi componenti hanno una struttura ed una filosofia d'impiego per alcuni aspetti profondamente diversa dai processori RISC di recente fabbricazione.

La caratteristica principale è rappresentata dalla flessibilità d'impiego in vari campi di applicazione, grazie alla possibilità di riconfigurare via software la topologia della rete.

Le principali caratteristiche e peculiarità di questi componenti saranno esaminate mentre per le specifiche tecniche riferirsi a [1], dopodichè verrà studiata in termini di prestazioni e costi una architettura di calcolo basata su " Transputer " per la ricostruzione di immagini 3-D ad ultrasuoni.

1. Prospettive dei computer ad alte prestazioni.

Con il termine supercomputers si intende in modo abbastanza generico alla classe di macchine di prestazioni più elevate disponibili ad un certo tempo. Le prestazioni sono valutate in termini di potenza di calcolo, larghezza di memoria, accessi I/O, interfacciamento con sistemi grafici, capacità del disco, collegamenti con banche dati. Molto spesso si parla di supercomputer anche quando una macchina ha elevate prestazioni solo

in alcuni dei campi sopra citati.

E' possibile fare una classificazione dei supercomputers in base al tipo di applicazioni sviluppate su di essi. Nella seguente tabella è riportato anche un fattore prestazioni/costo per le varie categorie.

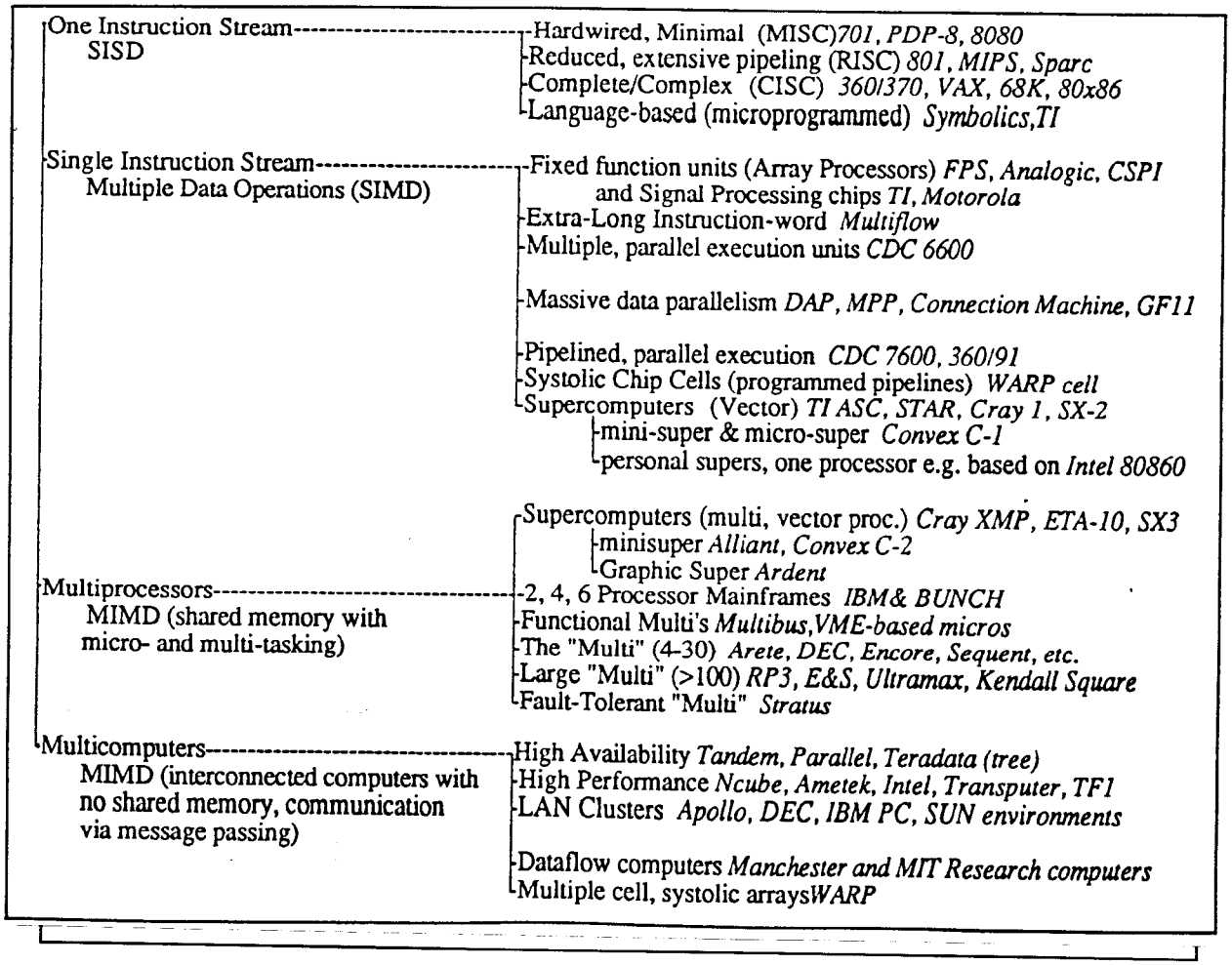
Tab. I Categorie di supercomputers e relativo fattore prestazione/costo [2].

Categorie	Fattore prestazioni/costo
1) Multiprogrammati - Programmi applicativi che risolvono classi generali di problemi	1
2) Monoprogrammati - Configurati per risolvere un applicazione specifica mettendo a disposizione tutte le risorse del sistema	10
3) Sistemi di calcolo costruiti per una applicazione specifica. Il software e l'hardware sono dedicati al problema	100 -> 1000

Molto spesso vengono confrontate le prestazioni dei supercomputers adottando dei programmi di prova chiamati " Benchmark " i quali stimano la potenza di calcolo in termini di operazioni in virgola mobile al secondo (flops), per esempio dei benchmark standard sono il Livermore Loops e il Linpack. Le prestazioni stimate con questi benchmarks non hanno un valore assoluto, perchè fino ad oggi tali programmi non riescono ad esprimere le prestazioni globali del processore; sono allo studio dei programmi di benchmark abbastanza generali che forniscono un indice di prestazioni globale e che possono essere applicati a processori basati su differenti architetture [8].

In tabella II sono riportate diverse strutture di calcolo mentre in tabella III alcuni dati relativi a nuove ditte

Tab. II Strutture dei supercomputers



costruttrici di computers ad alte prestazioni.
Dalla Tab. III si può notare che ancora non esiste una netta prevalenza di un tipo di computers rispetto ad un altro. Tuttavia emerge la tendenza all'impiego di sistemi di calcolo distribuito implementato sia con reti di computers, sia con architetture multiprocessore.

Sono ormai disponibili commercialmente una serie di microprocessori a basso costo ad alte prestazioni [3], basati su architetture RISC e tecnologie CMOS ed ECL (ad esempio un super-

Tab.III Ditte costruttrici di computer ad elevate prestazioni

Kind of computer	On market	Developing	Dead	Recent[1]
Supercomputers	2	2?	3	5
Mainframes (with vector proc.)	5	?	1	2
Mini-supers	5	2?	4	10
Graphic Supers	2	?	0	2
Total Supers	14	4	8	19
Array Processors	8	?	4	6
Massive Data Parallel	2	1?	?	3
Multiprocessors	6	2?	4	11
Multicomputers	16	?	2	18
Total (including supers)	46	7?	18	57
Superminis	7	1?	2	3
RISC-based computers	7	?	1	5

[1] Started after 1983.

PC con 4 processori da 20 mips ciascuno a bus condiviso costerà circa 80 milioni di lire nel 1990). In quest'ultimo caso si possono ottenere elevate prestazioni in termini di potenza di calcolo introducendo un elevato grado di parallelismo sia a livello dell'algoritmo, sia a livello dei microprocessori.

Per la scelta ed il progetto di un'architettura di calcolo parallelo questo rappresenta solo uno dei fattori da tenere in considerazione. Infatti, come si vedrà in seguito, per l'applicazione relativa alla ricostruzione 3-D esiste la necessità di concepire una struttura in cui i dati, memorizzati su disco, siano accessibili in breve tempo da tutte le unità di calcolo del sistema. Quindi le prestazioni di I/O risultano una specifica importante per questo tipo di applicazione.

Un grosso investimento sia economico, sia di risorse umane per la programmazione, viene fatto per le reti di computers. Il loro successo è stato determinato dallo sviluppo di efficienti e veloci reti di comunicazione per il passaggio dei messaggi fra i vari nodi della rete. I nodi sono connessi tra di loro per formare architetture chiamate " ipercubi ". Nella seguente tabella sono riassunte le principali caratteristiche di queste architetture di calcolo.

Tab. IV - Caratteristiche principali delle reti di computers (alias Multicomputers) :

1) Il loro impiego è rivolto alla monoprogrammazione
2) Numero di nodi : da 64 a 1024
3) Ogni nodo sostiene 10 mips con memoria di 4 Mbytes
4) Rapporto prestazioni/costo = 10 (vedi Tab. I)

Sono stati costruiti supercomputer per uso generale con architettura ad "ipercubo", ovvero un numero elevato di nodi connesssi tra di loro per formare un struuttura tridimensionale (vedi Fig. 1). Ad esempio in Europa sono realizzati multicomputers basati su i transputer, i quali sono dei processori ad alte prestazioni dotati di memoria propria e 4 link seriali full duplex a 20 Mbit/s per la comunicazione con altri transputer. Ogni transputer costituisce un nodo della rete la quale può essere facilmente riconfigurata via software tramite matrici di collegamento integrate su chip (chiamate crossbar switch).
La facilità di riconfigurare via software la topologia della rete di transputer viene sfruttata vantaggiosamente per la

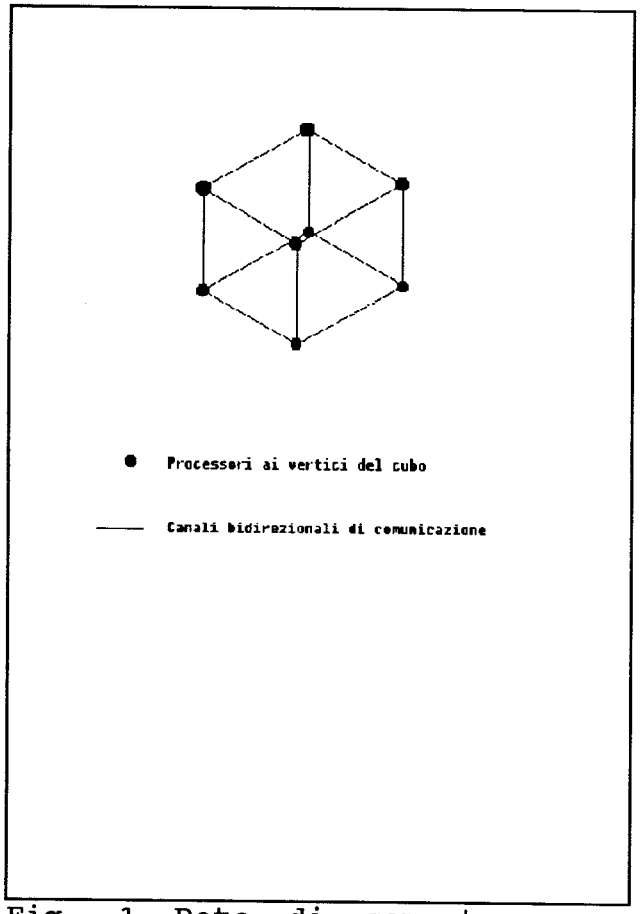


Fig. 1 Rete di computers con struttura ad "ipercubo"

soluzione dei problemi di fault-tolerance. La Fig. 2 mostra come si può risolvere il problema del fault-tolerance utilizzando due links del transputer; con essi si può escludere automaticamente il transputer guasto adiacente ed attivandone un altro o ridistribuendo il lavoro sui rimanenti [11].

La soluzione basata su transputer risolve i problemi relativi alla complessità di calcolo per applicazioni specifiche con un rapporto costo/prestazioni competitivo rispetto ad altre soluzioni[1]. Il grande interesse per questo tipo di architetture di calcolo è dimostrato dalla quantità di lavori che sono stati presentati negli ultimi anni [4]; i campi di applicazione sono i seguenti :

1) Elaborazione dei segnali (DSP) per applicazioni in astronomia, biomedica, riconoscimento di forme, previsioni metereologiche, sismica,
2) Controlli automatici per industrie spaziali,
3) Simulazioni di trasformazioni fluido dinamiche,
4) Soluzioni di problemi al contorno per campi elettromagnetici,
5) Computer graphics.

A questo punto è nesessario fare alcune osservazioni sulle caratteristiche dei transputer esposte precedentemente:
1) anche se si sono raggiunte alte prestazioni con i super nodi di transputer (190 mflops con 128 transputer T800-20 al STSC di Southampton [4]) queste strutture di calcolo non sostituiscono ancora i tradizionali supercomputers per le elevatissime prestazioni (1000-10000 mflops),

---

[1]In [4], "Adaption of an engine computational fluid dynamics code to a transputer based concurrent computer",pp.65 è stato presentato un lavoro dal prof. J.C. Dent in cui si dimostra come l'implementazione di un algoritmo su una rete di 4 transputer risulta più economica a parità di prestazioni con altri super mini computers o mainframe.

2) per ottenere le migliori prestazioni si deve programmare in linguaggio OCCAM, il costo della programmazione in termini di

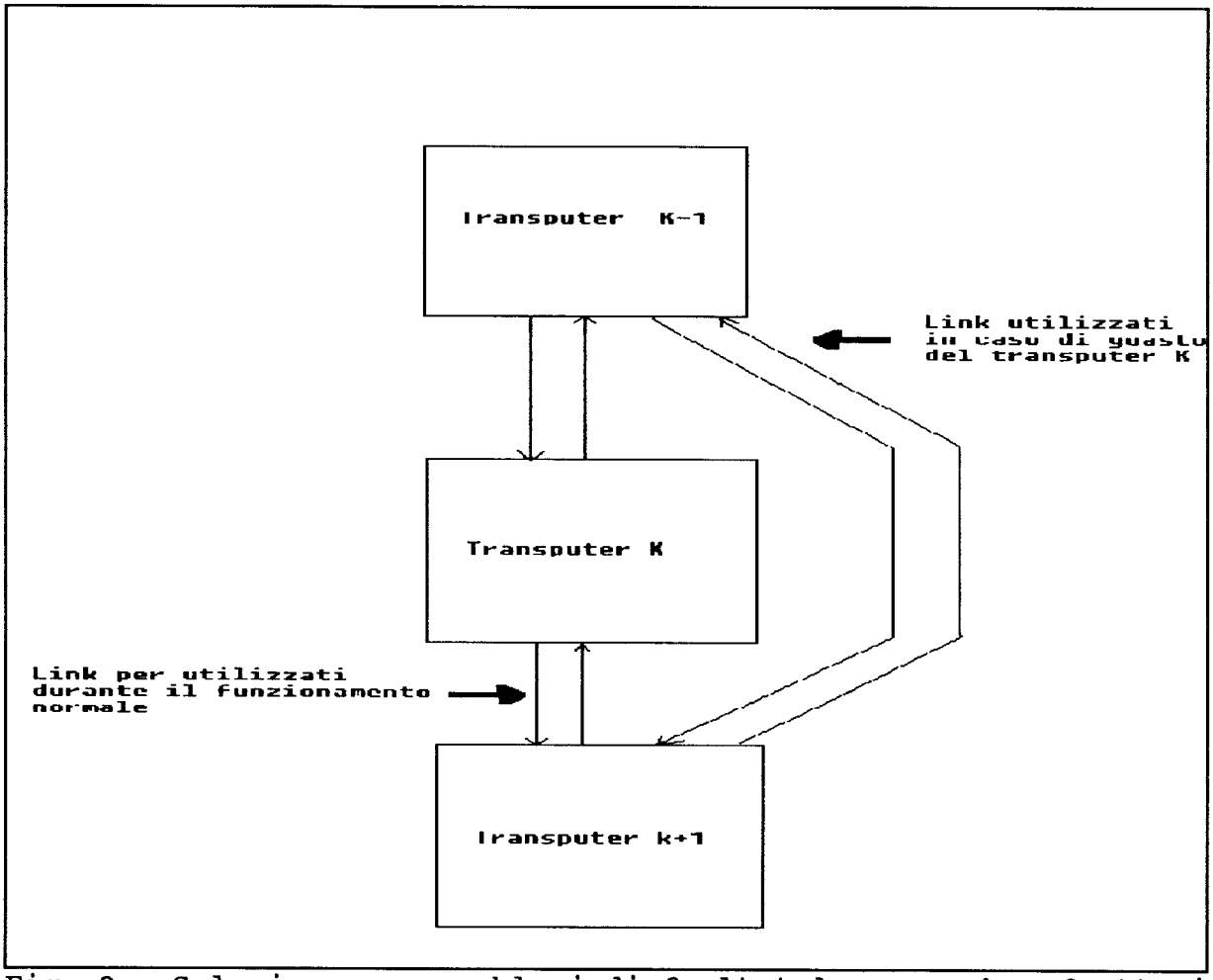


Fig. 2 - Soluzione per problemi di fault-tolerance che sfrutta i links del transputer

tempo/uomo è un fattore non ancora quantificabile, poichè molte applicazioni software sono state sviluppate in ambienti di ricerca come le Università, dove la forza lavoro è rappresentata in maggior parte dalla popolazione studentesca e dei ricercatori.

3) In Europa, a differenza degli Stati Uniti, non si ha una così larga diffusione dei collegamenti in rete con i supercomputers e

le soluzioni con reti interne risultano più comuni.
Nei successivi paragrafi verrano illustrati gli aspetti generali dei transputer, con particolare attenzione ai problemi relativi alla loro applicazione per la ricostruzione 3-D.

2. Aspetti generali delle reti di transputer.

Il transputer, prodotto dalla Inmos, fu introdotto nel mercato come un componente per alcuni aspetti rivoluzionario, in quanto la sua architettura interna ed il tipo di reti che si possono realizzare con esso, lo differenziano ancora oggi dagli altri processori. Proprio per questa sua particolare architettura si deve tenere presente che non è possibile un confronto diretto con le prestazioni di altri microprocessori. Un criterio più adatto per fare dei confronti sembrerebbe quello di paragonare le prestazioni relative all'intera rete di transputer, comprensiva delle periferiche per l'I/O, includendo anche il software sviluppato generalmente con il linguaggio ad alto livello OCCAM, il quale è stato inventato appositamente per il transputer, e in senso lato per la gestione dei processi concorrenti[2] . Quindi anche il

[2]Definizione : si chiamano in generale "processi concorrenti" una serie di processi definiti da differenti procedure, i quali sono eseguiti in paralello da una o più unità di calcolo. In generale tali processi pur essendo indipendenti, devono comunicare tra di loro, questo implica che esistono dei canali di comunicazione (soft. & hard.) gestiti da una serie di istruzioni software per la sincronizzazione della comunicazione.

linguaggio OCCAM non si può paragonare direttamente con le versioni real-time dei linguaggi più comuni come il Fortran, il Pascal o il C, poichè è provvisto di una serie di istruzioni relative alla gestione dei processi concorrenti che sono direttamente interpretate dal transputer con ovvi vantaggi di semplicità di programmazione e velocità di esecuzione. Alla base di questa filosofia vi è il concetto di sviluppare un linguaggio adattato alle caratteristiche hardware del processore, riuscendo così a sfruttare al massimo le sue prestazioni. Infatti nella Fig. 3 è illustrato il vantaggio offerto dalla soluzione transputer+OCCAM, che rappresenta la fusione delle tre fasi di progetto di un sitema di calcolo parallelo.

Nella valutazione del transputer per il suo impiego in applicazioni specifiche si deve considerare il problema dell'interfacciamento con l'esterno. In commercio sono disponibili schede di interfacciamento per periferiche standard, come controllori di bus GPIB, interfaccia SCSI per hard disk, frame grabber, schede per grafica ad alta risoluzione [9]. Invece per il progetto dell'hardware dedicato a periferiche specifiche si devono utilizzare i segnali di interfaccia della memoria, ne consegue che lo scambio dati tra il transputer ed il mondo esterno può essere realizzato con una memoria condivisa o con una ram dual-port.

3. La programmazione del Transputer in OCCAM : caratteristiche principali e confronti con altri linguaggi ad alto livello.

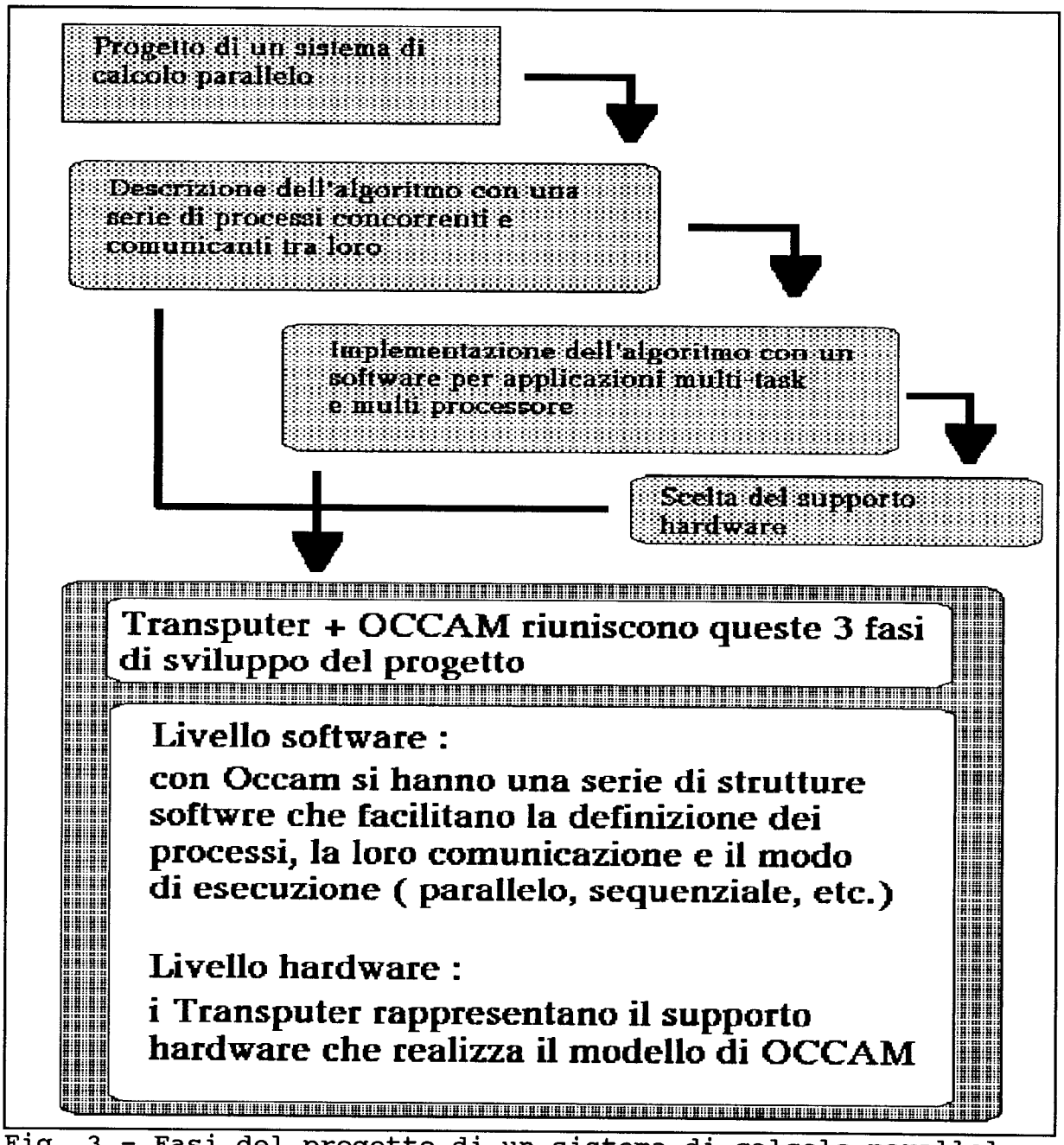


Fig. 3 - Fasi del progetto di un sistema di calcolo parallelo

Il linguaggio OCCAM rappresenta la scelta ideale per programmare il transputer, poichè il transputer realizza in hardware il modello di OCCAM. L'introduzione di questo linguaggio ha risolto in modo semplice ed efficiente il problema della programmazione di processi concorrenti. Infatti la struttura classica dei calcolatori basata sul modello di calcolatori di von Neumann (singola CPU che colloquia con una singola memoria) presenta limiti pesanti per questo tipo di applicazioni, mancando veloci canali di comunicazione hardware e sopratutto linguaggi di programmazione che siano dotati di "istruzioni ad alto livello" per accedere a tali canali di comunicazione. Ad esempio le istruzioni "IN" e "OUT" di OCCAM servono per ricevere e trasmettere un dato su un canale di comunicazione specificato.

Un vantaggio di OCCAM è quello di poter definire il proprio algoritmo come una serie di processi concorrenti comunicanti tra di loro, la cui esecuzione può essere affidata in fase di messa a punto del programma ad un singolo transputer. In questo modo il programma può essere provato e corretto con l'aiuto di un debugger e l'esecuzione dei vari processi sarà controllata dallo "scheduler" microcodificato del transputer. Supponendo che i tempi relativi alla comunicazione dei processi siano trascurabili, si può in teoria avere un incremento lineare della velocità di calcolo[3],

---

[3] Il concetto di "linearità" nei sistemi di calcolo parallelo si riferisce alla relazione che esiste tra numero di processori e la velocità del sistema [10]

assegnando l'esecuzione di ogni processo ad un transputer, e trasformando i canali di comunicazione software con i link seriali hardware del transputer vedi Fig. 4.

Il passaggio da uno a più transputer risulta estremamente poco costoso in termini di programmazione, in quanto basta cambiare le poche istruzioni relative alla definizione dei tre processi e dei canali di comunicazione. Tuttavia in campo pratico il problema del della comunicazione dei processi e quindi la loro sincronizzazione non è facilmente valutabile a priori, questo può allontanare molto le prestazioni in termini di velocità dall'andamento lineare teorico.

Per le applicazioni real-time OCCAM offre alcune interessanti caratteristiche. La programmazione real-time richiede in fase di messa a punto del programma la simulazione degli eventi esterni, i quali possono essere definiti facilmente come processi OCCAM, temporizzati dai Timer del transputer ad alta e bassa risoluzione (rispettivamente 1 e 64 μs). Sempre per le applicazioni real-time è spesso indispensabile una assegnazione dinamica delle priorità dei processi, i quali normalmente sono "schedulati" con il metodo time-slice [5]. Ad esempio con l'istruzione "PRI" di OCCAM si può modificare la priorità di un processo. Tale istruzione ad alto livello viene direttamente decodificata ed eseguita dal transputer.

L'impiego di sistemi multiprocessore per incrementare le prestazioni ha bisogno di adeguati linguaggi ed ambienti di sviluppo del software. Purtroppo si deve notare che anche nel caso delle reti di transputer, come altri sistemi multiprocessori, non

sono ancora assistiti da efficienti programmi di debugger [6]. Esistono diversi tipi di debugger per il transputer, ma essi si basano ancora sull'idea delle finestre nelle quali sono riportate parti di testo del programma. Attualmente il panorama dei programmi di debugger per sistemi multiprocessore vede l'affermarsi di tre differenti tipi di debuggers [6], i quali cercano di risolvere i problemi classici come la gestione di molti processori (100 -> 1000), la sincronizzazione, il rallentamento dell'esecuzione. Per il transputer ancora non esistono programmi di debugger in cui le informazioni relative ai nodi della rete sono

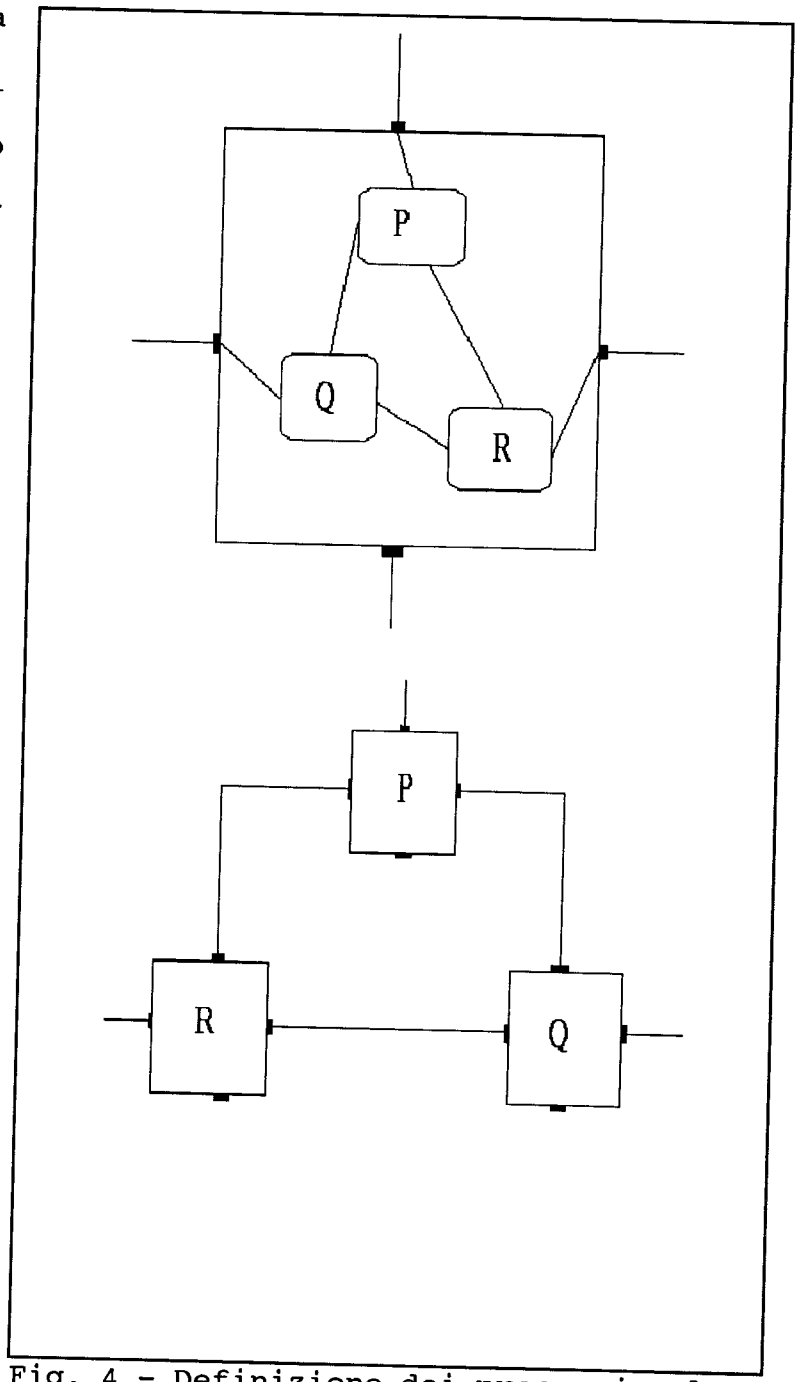


Fig. 4 - Definizione dei processi e loro implementazione su singolo transputer (alto) e su più transputer (basso)

rappresentate in forma grafica ed in modo compatto in modo da monitorare l'intero carico di lavoro relativo a tutta la rete ed eventualmente risolvere i problemi di equilibrio del carico di lavoro.

Il linguaggio OCCAM riguardo al transputer può essere considerato il suo linguaggio assembly, anche se OCCAM è caratterizzato da una sintassi molto evoluta, molto vicina ai linguaggi di programmazione ad alto livello. Esiste anche un vero assembly del transputer ma è inutilizzabile per la programmazione di sistemi multiprocessore, tale linguaggio viene disassemblato solo in certe fasi di debugging.

In confronto agli altri linguaggi l'OCCAM ha alcune limitazioni. La prima è l'allocazione statica della memoria (come il Fortran) che non permette l'esecuzione dei programmi ricorsivi come in Pascal. La seconda è la mancanza di strutture come lo STACK, la cui gestione richiederebbe una sistema operativo abbastanza complesso quando si eseguono molti processi concorrentemente. La mancanza di queste due caratteristiche è stata una scelta di progetto la quale ha privilegiato altre caratteristiche come la semplicità e la modularità del software, le prestazioni (si hanno tempi di context-switching[4] di 1 μs contro 100 μs con UNIX), la sicurezza del software applicativo

[4] Definizione : le prestazioni di context-switching si riferiscono al tempo necessario per passare da un task ad un altro. Questo parametro insieme al tempo di latenza dell'interrupt caratterizza le prestazioni dei microprocessori per applicazioni multi-task

(probabilità di errori run-time). Sulla sicurezza si può dire che grazie alla allocazione statica della memoria la maggior parte degli errori si possono correggere in fase di compilazione, riducendo così i più insidiosi e nascosti errori run-time.
Per questo il transputer è stato impiegato in molte applicazioni dove il parametro sicurezza-prestazioni è determinante, come ad esempio nel controllo automatico di vettori spaziali.
Concludendo il transputer si può considerare come un componente programmabile tramite OCCAM per l'esecuzione di processi concorrenti. OCCAM garantisce una programmazione efficiente e sicura (nel senso degli errori) con una sintassi chiara e semplice per questo tipo di applicazioni.

4. Considerazioni di progetto di un sistema di calcolo per la ricostruzione dinamica 3-D ad ultrasuoni basato su transputer.

In questa seconda parte del report, saranno esposte le caratteristiche generali di un sistema di ricostruzione tridimensionale del cuore basato su ecografie ad ultrasuoni, ed i problemi relativi al progetto di una architettura di calcolo parallelo per poter ottenere da queste informazioni una visione del cuore in movimento in tempo quasi reale.
Quindi il nostro problema e' composto da :

* una struttura hardware che ha il compito di acquisire e memorizzare i dati (ecografie secondo vari piani ed in diversi istanti del ciclo cardiaco),

** degli algoritmi di ricostruzione (interpolazione 3-D e ray-tracing modificato) gia' sviluppati con linguaggi ad alto livello

[12],

*** Problema : attualmente il tempo per la ricostruzione del cuore in movimento secondo un certo punto di vista richiede tempi di calcolo eccessivamente lunghi per poter pensare ad un suo impiego come esame di routine,

**** Obiettivo : progetto di una rete di transputer per poter accelerare il processo di elaborazione delle immagini (visione dinamica del cuore in tempo quasi reale) e stima delle prestazioni e dei costi.

4.1. Descrizione del processo di ricostruzione.

In Fig.5 e' descritto lo schema del processo di ricostruzione. Con una speciale sonda rotante ad ultrasuoni vengono acquisite le ecografie secondo vari piani con lo stesso asse principale della sonda ma ruotati di un certo angolo ϕ. L'angolo ϕ e' sufficientemente piccolo (1.8° - 3°) per garantire una buona risoluzione su tutto il volume ricostruito. Per ogni piano si ha una acquisizione sincronizzata con il segnale ECG di un certo numero di ecografie entro lo stesso ciclo cardiaco per la ricostruzione dinamica del cuore. Il tempo di acquisizione e spostamento del piano di scansione e' circa 1s. Generalmente dai 16 ai 25 frames al secondo sono sufficienti a descrivere il movimento del cuore.
Nel caso in cui si voglia la massima risoluzione, il processo di acquisizione con 25 frames ed una risoluzione angolare di 1.8° richiede un tempo di 100 s, in quanto i piani dopo l'angolo uguale a 180° si ripetono. Alla fine del processo di acquisizione e memorizzazione si hanno 100 blocchi di 25 immagini ciascuno, contenenti le ecografie relative ai 180 piani con cui viene campionato il volume, e relativamente ad ogni piano, le 25

ecografie che descrivono nel tempo il movimento del cuore su quel piano.
Il sistema hardware puo' essere programmato via software e permette di variare l'angolo ϕ, il numero di frames al secondo, eseguire un controllo sul tempo medio del ciclo cardiaco in modo da segnalare eventuali scostamenti dalle condizioni normali. Lo stesso programma di gestione dell'hardware provvede a memorizzare in tempo reale i blocchi di immagini ecografiche su memorie di massa.

4.2. Stima della quantita' dei dati.

Uno dei problemi di questa applicazione e' la grande massa di dati che si deve memorizzare su supporti di memoria non volatili che poi deve essere elaborata da una architettura di calcolo parallelo.
La dimensione dei blocchi di immagini e' legata al passo di campionamento con cui si campiona lo spazio ed il tempo.

Tab. V - Dati per il caso con massima risoluzione

Ecografie 512 X 512 Pixel
25 frames al secondo
256 livelli di grigio (8 bit) per pixel
Passo angolare ϕ = 1.8 ° => 100 Piani

Con i dati della precedente tabella possiamo valutare la dimensione dei dati e quindi della memoria :

Occupazione di memoria = 512 X 512 X 25 X 100 = 655.36 Mbyte

Con questi valori dei dati il supporto di memoria non puo' essere altro che un nastro magnetico oppure dei dischi ottici riscrivibili collegati al personal computer. Tuttavia il caso con la massima risoluzione analizzato precedentemente rappresenta solo il limite superiore delle prestazioni del sistema.
Accettando una minore risoluzione, ma sempre sufficiente a

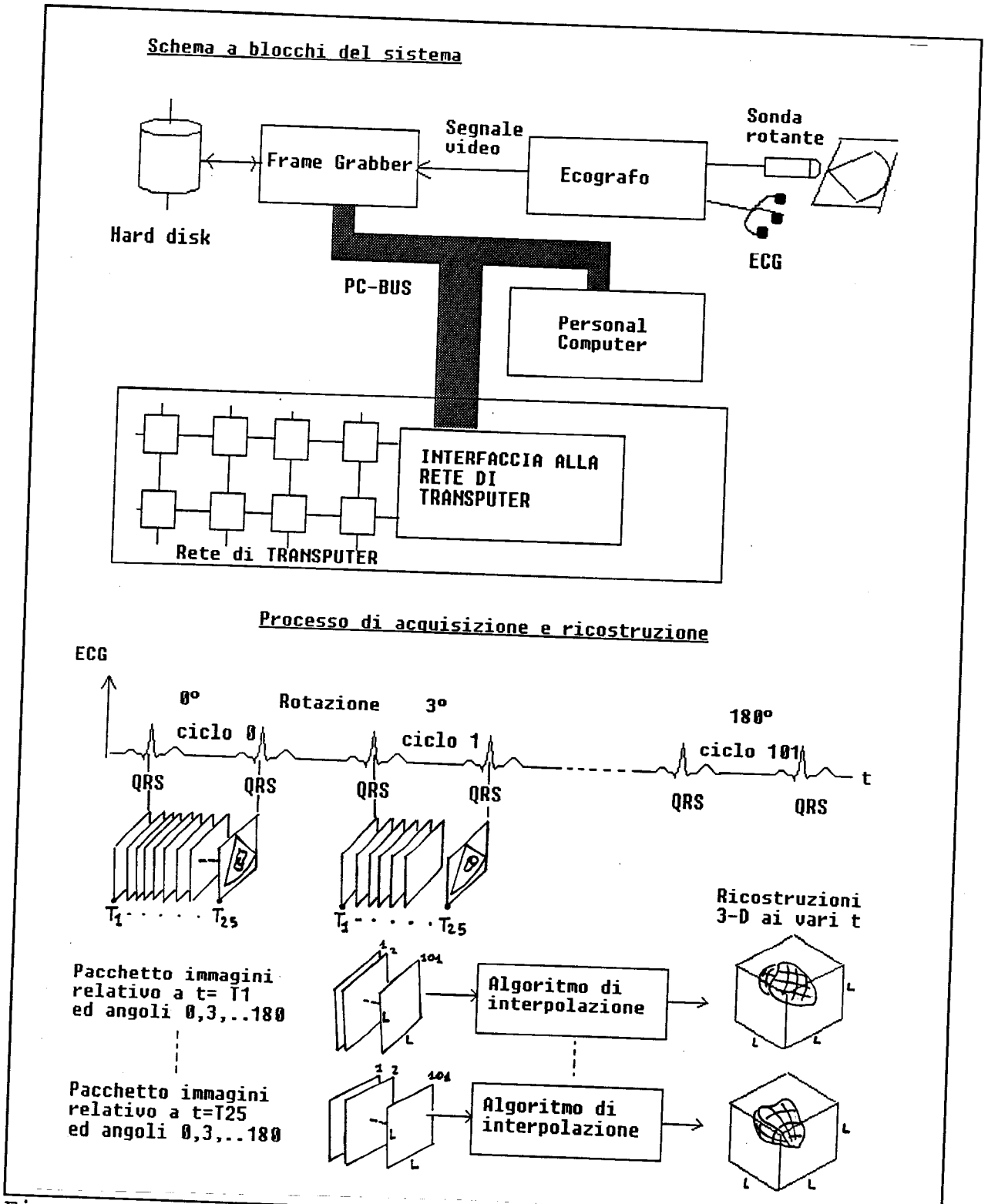


Fig. 5 - Schema generale del processo di acquisizione e memorizzazione delle immagini ecografiche.

descrivere il movimento del cuore, si puo' considerare per la stima della occupazione di memoria i dati nella seguente tabella :

Tab. V bis - Dati per il caso con media risoluzione
Ecografie 256 X 256 Pixel 16 frames al secondo 256 livelli di grigio (8 bit) per pixel Passo angolare ϕ = 3 ° => 60 Piani

In questo caso si ottiene :
Occupazione di memoria = 256 X 256 X 16 X 60 = 62.914560 Mbyte
Tale valore risulta di circa un ordine di grandezza inferiore al caso precedente e quindi si ha un notevole risparmio sulla occupazione di memoria oltre che un minore tempo di elaborazione. L'interpolazione lineare ha lo scopo di trasformare le coordinate cilindriche dei pixel delle immagini ecografiche (le immagini cosi' acquisite formano una stella di piani con asse quello della sonda) in coordinate cartesiane su una matrice cubica con dimensioni, in questo caso, di 256 x 256 X 256 = 16.777216 Mbyte. Con 16 frames al secondo avremo cosi' 16 matrici cubiche sulle quali si potra' applicare l'algoritmo di ray-tracing per avere una visione 3-D del cuore in movimento, secondo diversi punti di vista. Alla fine dell'interpolazione le 16 matrici cubiche occuperanno uno spazio totale di memoria pari a 16.777 X 16 = 268.43 Mbyte.

4.3. Soluzione con la " Transputer Farm ".

Nella nostra applicazione si vuole cercare di accelerare il processo di interpolazione e di ray-tracing portando tali programmi, che attualmente sono eseguiti su macchine con singolo processore, su una architettura di calcolo parallelo realizzata con i Transputer, tale soluzione si è dimostrata efficace in problemi di elaborazione di immagini mediche 3-D [15].

La parallelizzazione di una specifica applicazione e' un problema abbastanza complesso quando la sua implementazione prevede una serie di processi tra di loro comunicanti, nel senso che l'esecuzione di ognuno non e' indipendente dalle altre, e quindi si deve avere una sincronizzazione del sistema di processori per lo scambio di dati e risultati in fasi intermedie del programma.

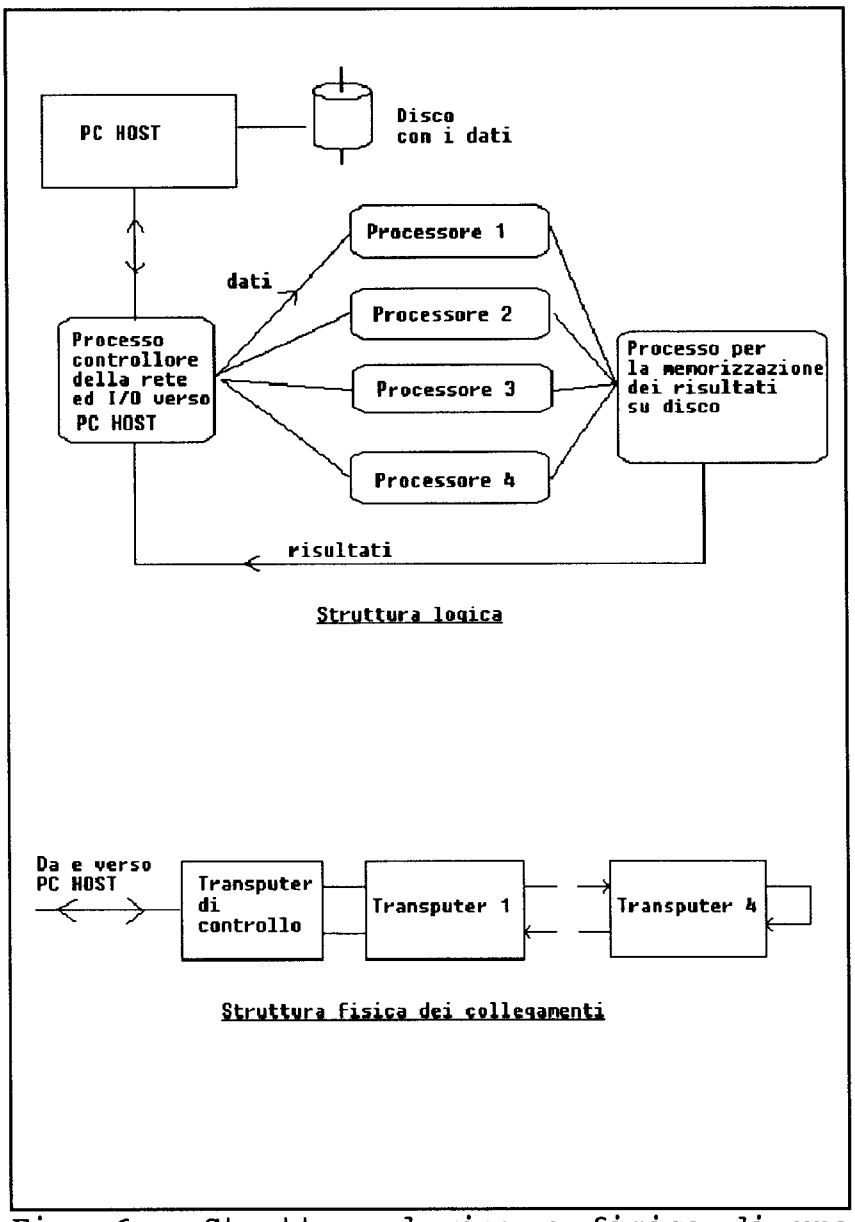


Fig. 6 - Struttura logica e fisica di una "Transputer Farm"

Nel nostro caso siamo facilitati in quanto e' possibile la parallelizzazione del calcolo a livello di algoritmo, ovvero uno stesso programma puo' essere copiato su ogni transputer, ed i dati da elaborare possono essere suddivisi in sottoinsiemi indipendenti che vengono distribuiti ai vari transputer. Alla fine i risultati dei singoli transputer vengono inviati ad un controllore che provvede alla loro memorizzazione sul disco. Questo tipo di parallelizzazione e' al

livello piu' basso di granularita', ed e' facilmente realizzabile con una "Trasputer Farm"[5], la cui struttura logica e fisica e' mostrata in Fig.6. Nella figura sono stati riportati 4 processori come esempio.

Un altra caratteristica della nostra applicazione e' quella che si tratta sostanzialmente di programmi di calcolo che fanno uso solo di funzioni standard del linguaggio C con il quale sono stati sviluppati. Questo facilita la loro portabilita' su i transputer tramite un opportuno compilatore per C parallelo. Nei prossimi paragrafi verranno analizzati alcuni risultati sperimentali, per vedere la perdita di prestazioni dovuta alla "traduzione" di un programma in C su un transputer rispetto alla soluzione di riscrivere in OCCAM tutto il programma.

In Fig. 6 e' stata scelta come topologia della rete una cortina di transputer. La soluzione topologica alternativa alla cortina di transputer potrebbe essere quella ad "ipercubo". Questa soluzione avrebbe il vantaggio di diminuire la lunghezza dei percorsi dei dati ma il processo che controlla la comunicazione sarebbe essere piu' complesso dovendo gestire piu' canali di comunicazione (con l'ipercubo si utilizzano 4 links per ogni transputer). La portabilita' di un programma gia' esistente su una "transputer farm" deve essere caratterizzata da un breve tempo di sviluppo del software di comunicazione per la gestione dei dati nella rete e per ragioni di affidabilita' il piu' semplice possibile. Per questa ragione una cortina di transputer rappresenta una buona soluzione quando non vi siano particolari esigenze di velocita' di trasferimento dati, inoltre tale topologia puo' essere facilmente modificata con una struttura ad anello connettendo il primo e l'ultimo transputer; quest'ultima topologia si e' già rivelata efficiente per l'implementazione del ray-tracing [10].

---

[5] Il termine "Transputer Farm" (traduzione letterale Fattoria di Transputer) significa che una serie di transputer sono programmati per fare lo stesso lavoro, ovvero eseguono tutti lo stesso programma ma su dati iniziali differenti.

4.4. Descrizione dei moduli software per la gestione della rete.

In una "transputer farm", i moduli software necessari a gestire lo scambio dei dati tra i transputer sono di carattere generale ed un primo nucleo di programma scritto in OCCAM si trova gia' sviluppato in diverse pubblicazioni tecniche [11]. In Fig.7 sono riportati i principali moduli software.
Per non degradare eccessivamente le prestazioni dei transputer, nel progetto del software si deve tenere conto che i processi di comunicazione sono eliminati dallo scheduling finche' la comunicazione e' in atto, lasciando cosi' la CPU e FPU del transputer a disposizione per i calcoli della programma principale. La CPU viene interrotta solo all'inizio o alla fine della comunicazione su un link, questo implica che e' meglio trasferire grandi pacchetti di dati, compatibilmente con la memoria RAM disponibile su ogni transputer. Dato che ogni pacchetto di dati deve avere una propria collocazione (indirizzo di memoria del vettore contenente i dati) ed una codifica per la sua identificazione (ad esempio l'angolo del piano di appartenenza dell'ecografia nella nostra applicazione) e' necessario massimizzare il seguente rapporto :

E = Lunghezza dati/Lunghezza totale pacchetto

Considerando un immagine 256 x 256 = 65536 byte di dati ed una decina di byte per la sua descrizione abbiamo una efficienza E elevatissima E = 65536/(65536+10) = 0.99984 .

Il modulo controllore IN\OUT sul transputer di controllo ha il compito di creare un buffer di ingresso e di uscita rispettivamente con le immagini da elaborare prelevate da disco e con le immagini gia' elaborate da memorizzare su disco. Inoltre inizializza la configurazione della rete e carica i rispettivi programmi. Inoltre quando un transputer ha terminato la elaborazione dei suoi dati invia i risultati al transputer di controllo e richiede altri dati al processo router IN. Con questa

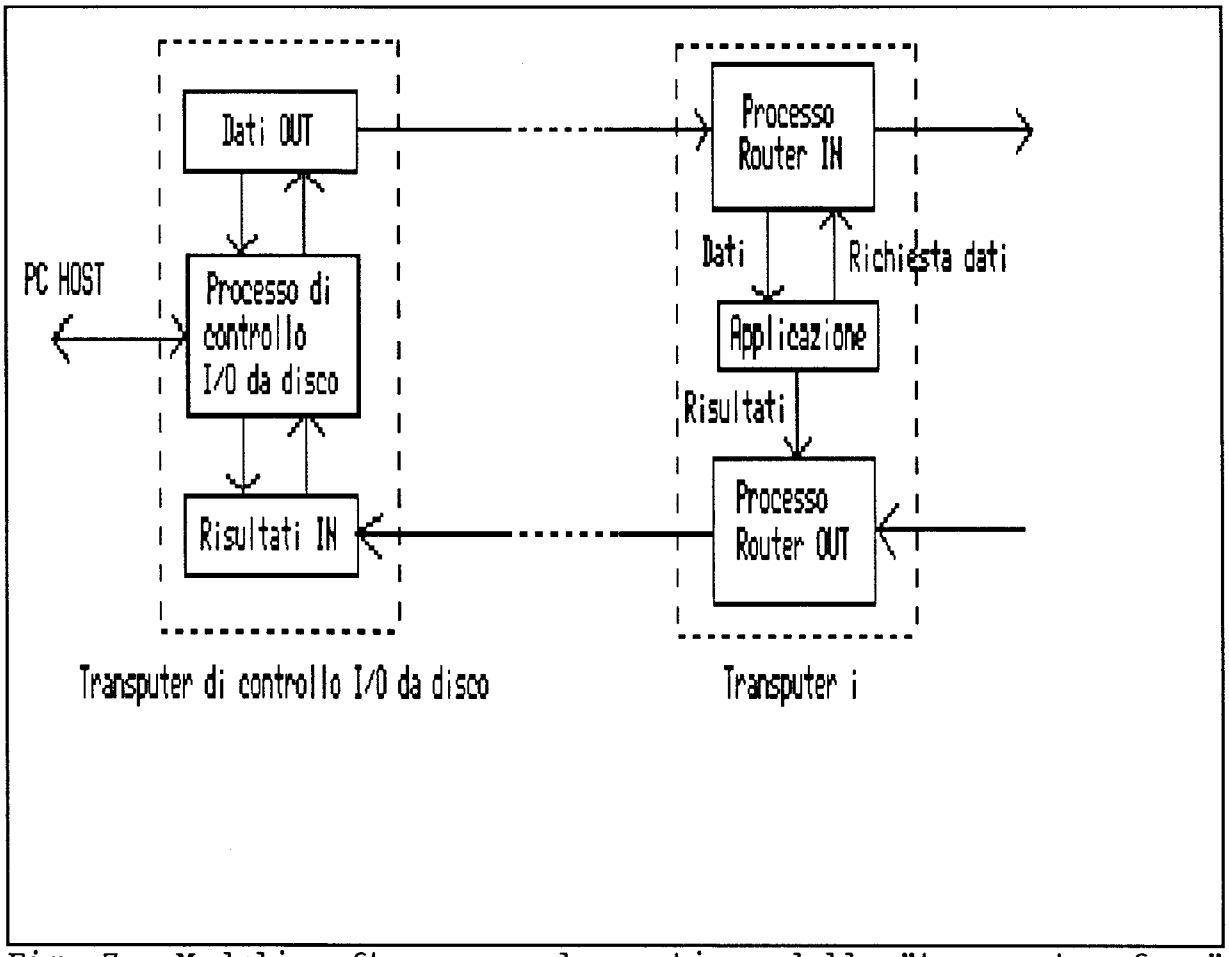


Fig. 7 - Moduli software per la gestione della "transputer farm"

tecnica molto semplice di autobilanciamento, si ha un carico di lavoro ugualmente distribuito tra tutti i transputer.

4.5. Prove con programmi di "Benchmark" per la stima delle prestazioni dei transputer.

In questo paragrafo sono riportati i risultati ottenuti da un serie di prove eseguite sui transputer per stimarne le prestazioni dal punto di vista di elaborazione di un singolo processo su un singolo transputer. Oltre a cercare una conferma delle prestazioni dichiarate dalla INMOS si puo' cosi' dare una stima di un fattore di merito rispetto a delle macchine di riferimento. Il problema della significativita' dei programmi di benchmark e' ancora oggi

un problema aperto se si pretende di avere una indicazione delle prestazioni globali della macchina. Nel nostro caso dobbiamo confrontare una specifica applicazione su una serie di computers, quindi i programmi messi a punto sono mirati ad evidenziare le prestazioni del transputer con tale applicazione rispetto alle altre macchine.

I programmi di interpolazione e di ray-tracing che utilizziamo fanno uso di aritmetica in virgola mobile, oltre ad operare somme, prodotti, confronti, tra le componenti di grandi matrici di dati tridimensionali. Per questo scopo sono stati messi a punto i seguenti programmi:

1) Anello di Livermore con numero iterazioni N=10000
2) Floating Point Bench con N = 25000
3) Setaccio di Eratostene dei numeri fino a 8190 e con N=150

Su personal computer i programmi sono stati sviluppati in linguaggio C (QC V 5.0 Microsoft) con ottimizzatore della velocita' in compilazione, mentre sul transputer in OCCAM 2. Premesso questo nelle seguenti tabelle sono riportati i tempi di esecuzione per le varie prove.

Tab. VI - Tempi di esecuzione con 3 diversi benchmarks

Tipo di macchina	Tempo di esecuzione [centesimi di sec]		
	Anello di Livermore	Floating Point Bench	Setaccio di Eratostene
IBM AT i286	253	3345	1872
IBM PS2/80 i386 clock 20 MHz	60	1301	820
COMPAQ 33 i386 clock 33 MHz	22	472	308
Transputer T800-20 clock 20 Mhz	11	411	924

Nella seguente tabella e' riportato il tempo di esecuzione per vari valori di N dell' "Anello di Livermore",su un transputer T800-20. Il tempo e' stato calcolato con il timer ad alta risoluzione (1μs). In questo modo si puo' vedere se le prestazioni per un singolo processo sono lineari rispetto alla dimensioni del problema.

Tab. VII - Tempi di esecuzione su tranpsuter T800-20 dell'Anello di Livermore

N	Tempo di esecuzione [μs]	Speed Up	Errore di linearita' %
1000	10721	1	2.6
10000	110176	10.27	0.00617
25000	275426	25.69	0.00108
50000	550846	51.38	0

In tabella VIII sono riportati i risultati ottenuti con il programma Floating Point Bench per diversi valori di N, con 2 diversi transputer, e con i programmi scritti in OCCAM ed Helios C.

Tab. VIII - Tempi ed errori del Floating Point Benchmark su due differenti transputer e con due diversi linguaggi

	N = 2500		N = 25000	
	Helios C	OCCAM	Helios C	OCCAM
δN/N %	-1.2E-06	4E-11	-1.2E-06	-1.2E-08
Tipo di Transputer	Tempi di esecuzione [centes. di sec.]			
T800-20 clock 20 MHz	96	41	959	410
T415-15 clock 15 MHz senza Float. Point Unit	3265	1751	30600	17514

Infine in tabella IX sono riportati i risultati ottenuti con "Whetstone benchmark"[6], largamente accettato ed usato in letteratura. Tali dati sono stati ripresi da [13].

Tab. IX - Risultati del Whetstone Benchmark

Tipo di CPU	Migliaia di Whetstone per secondo	
	doppia prec.	singola prec.
IMS T800-20	2932	4548
i386 clock 20MHz con coprocessore matematico i387	1730	1860
i8086 clock 8MHz con coprocessore matematico i8087	152	178

Dalle precedenti tabelle e' possibile estrarre una fattore di merito F per il transputer T800-20, con il quale si puo' stimare il tempo di esecuzione di una singola applicazione portata su un T800-20, conoscendo il tempo di esecuzione su un altra macchina.

Da questa serie di prove si nota come le prestazioni del T800-20 sono confrontabili o superiori alle macchine di confronto.

Alcune osservazioni sui risultati ottenuti possono essere fatte:

1) Nella tabella VI il migliore fattore di merito si ottiene con l'Anello di Livermore, il cui programma in OCCAM e' stato concepito appositamente per esaltare la concorrenza tra CPU ed FPU del T800-20. Gli altri 2 programmi di benchmark sono stati scritti ex-novo senza particolari ottimizzazioni software, che puo' avere solo una programmatore gia' esperto di OCCAM, quindi rappresentano una

[6] Questo benchmark è composto da diversi moduli software, i quali esaltano singolarmente certe prestazioni del processore. Alla fine fornisce un indice globale di prestazione.

situazione reale di lavoro; essi mostrano prestazioni inferiori all'Anello di Livermore.

2) I test realizzati sono stati dimensionati in modo da operare in zona di linearita' tra dimensione e tempo di esecuzione, come si vede dalla tabella VII, in modo da avere stime attendibili dei tempi di esecuzione.

3) Dalla tabella VIII si vede che lo stesso programma scritto e compilato in C dimostra una degradazione delle prestazioni, in termini di tempo di esecuzione, rispetto ad OCCAM di un fattore 2, ed una perdita di precisione di 2 ordini di grandezza. Questo risultato si ripercuote nei casi in cui si vuole portare una applicazione gia' esistente con un certo linguaggio su un transputer, come nel nostro caso. Il problema della scarsa efficienza dei compilatori C parallelo per il transputer è già stato evidenziato da D. Pountain in [14]. Tuttavia si noti che il compilatore C usato nelle nostre prove non e' recente, percio' dovrebbero essere confrontate le ultime versioni disponibili dei compilatori C. Fra i tempi di esecuzione dei 2 transputer esiste circa un fattore 35 dovuto oltre ad una frequenza di clock maggiore, alla presenza di una F.P.U. sul T800-20.

4) La stima del fattore F rispetto ad un i386 + i387 con clock 33 MHz di un COMPAQ 33 si puo' ottenere con la media dei 3 fattori di merito ottenuti dalla tabella VI.

$$F = (22/11 + 472/411 + 308/924)/3 = 1.16$$

Questa stima del fattore di merito F rientra tra i due valori che si ottengono dalla tabella IX estrapolando le prestazioni a 33 MHz di un i386+i387. Con tali valori si ha F = 1.48 in singola precisione ed F = 1.03 in doppia precisione. Tenendo presente che la versione a 30 MHz del T800 e' gia' disponibile commercialmente si puo' sicuramente stimare :

$$F = 1.16 * 30/20 = 1.74$$

Considerando inoltre che nella nostra applicazione le operazioni di confronto logico (valutate dal Setaccio di Eratostene) sono molto inferiori rispetto alle operazioni aritmetiche, significa ch

la stima di F e' cautelativa rappresentando il caso peggiore, quindi ingeristicamente valida.

4.6. Analisi delle prestazioni verso il costo per il progetto della transputer farm.

Nel progetto di questa architettura di calcolo, e' interessante valutare l'influenza di alcuni fattori, come il numero di transputer, la velocita' di comunicazione, la memoria a bordo, sulle prestazioni ed il costo del sistema.

A tale scopo e' stata sviluppata una formula riassuntiva per il tempo totale di elaborazione (T.el.tot.) del programma di interpolazione lineare.

Definizione dei simboli.

K numero totale di piani da interpolare = (180°/ ϕ)*(Frames/s)
L*L dimensioni [byte] di un immagine ecografica (es. 512*512)
r velocita' del canale di comunicazione [Mbit/s] [7]
M memoria RAM a bordo di un transputer [8] [byte]
N numero di transputer impiegati
Te tempo di elaborazione per interpolare un immagine di dimensioni L*L su una certa macchina [s]
F fattore di merito del transputer rispetto ad una certa macchina

$$T.el.tot. = \left(\frac{M}{L*L} * \frac{Te}{F} \right) * \frac{K}{N} * \frac{L*L}{M} + N * \frac{M*8}{r}$$

[7] Con alcune prove sperimentali si e' stimata r al variare della lunghezza dei pacchetti di dati trasferiti. Per pacchetti >1000 bytes si e' ottenuto una stima cautelativa di r = 72 % della velocita' nominale del link. Con 20 Mbit/s, r = 20*0.72 = 14.4 Mbit/s

[8] La memoria deve essere al minimo capace di contenere il programma ed una immagine di dimensioni L*L

Osservazioni :

1) Nella formula precedente si e' assunto che il processo di comunicazione richieda la CPU per un tempo trascurabile rispetto al tempo di calcolo richiesto dall'algoritmo di interpolazione, ovvero che i due processi siano realmente paralleli. Sviluppando il primo termine si vede che il T.el.tot non dipende dalla dimensione della memoria a bordo e dalle dimensioni delle immagini.

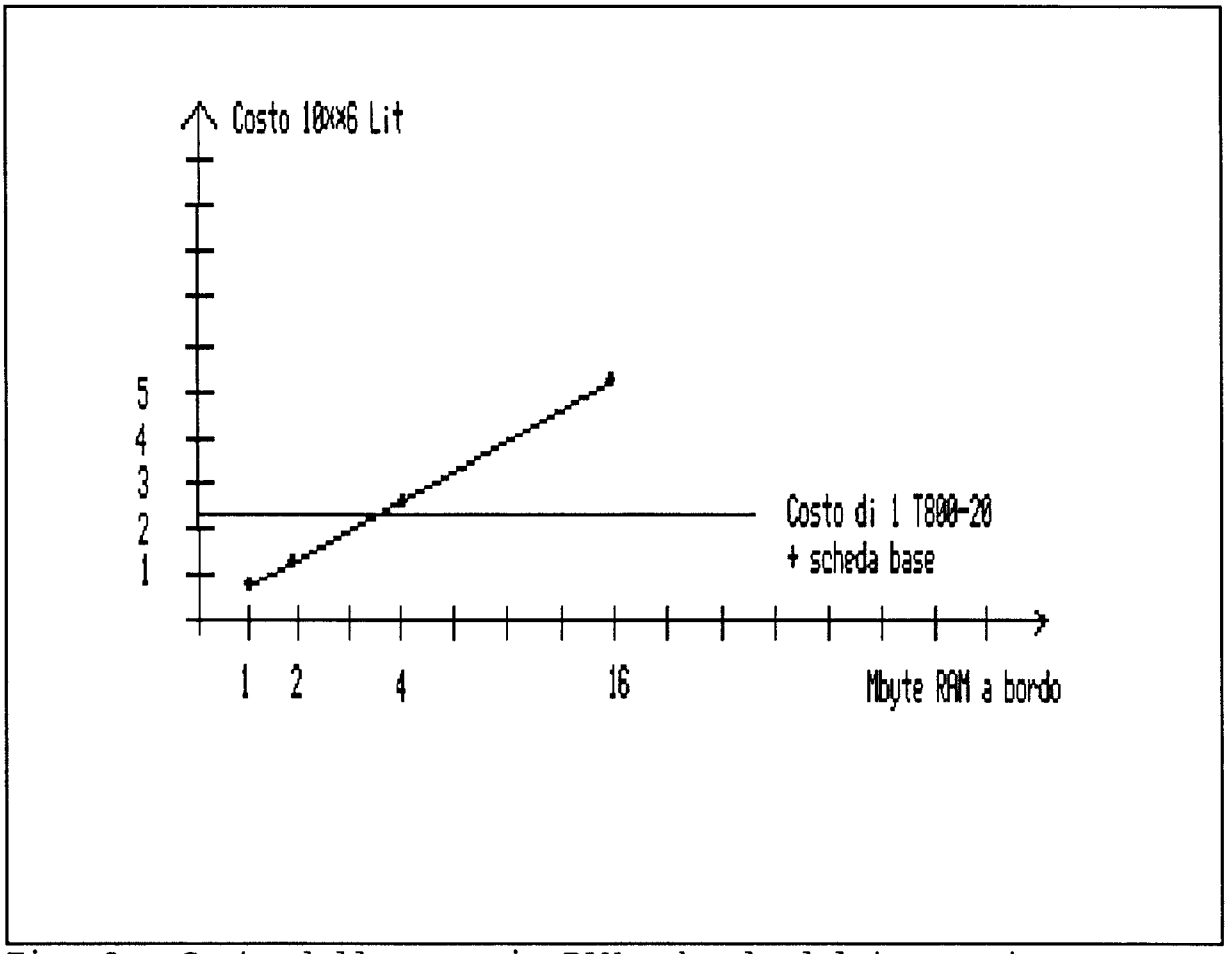


Fig. 8 - Costo delle memorie RAM a bordo del transputer

In effetti con una memoria piu' grande si possono caricare piu' immagini e quindi diminure il numero di richieste alla CPU per il controllo della comunicazione, migliorando cosi' il rapporto fra il tempo di calcolo ed il tempo di comunicazione. Tale rapporto e' ulteriormente migiorato da valori piu' grandi di r. Si deve tenere

presente che il costo di memorie RAM veloci a bordo del transputer e' lineare con le dimensioni della memoria e puo' facilmente superare il costo del singolo transputer (vedi Fig. 8).

2) Con le ipotesi precedenti dalla formula del T.el.tot si vede che le prestazioni sono lineari con il numero di transputer utilizzati. Un criterio per la scelta del numero di transputer potrebbe essere quello di valutare il numero di transputer in modo da non superare il limite dato dai canali di comunicazione. Infatti aumentando il numero di transputer si potrebbe raggiungere la situazione in cui alcuni di essi rimangono in attesa per l'accesso al canale di comunicazione (vedi Fig.9). A tale scopo si puo' utilizzare la seguente formula di progetto :

$$N \leq \frac{Te}{F} * \frac{M}{L*L} * \frac{r}{M * 8}$$

Anche in questo caso la dimensione della memoria non influisce sul numero di transputer.

3) Il secondo termine del T.el.tot. rappresenta un tempo di inizializzazione di tutti i transputer in cui si caricano per la prima volta i dati da elaborare. Questo termine è trascurabile se la memoria a bordo è piccola poiché significa che vengono trasferiti pochi dati alla volta.

5. Conclusioni e sviluppi futuri.

Dall'analisi condotta in questo report risulta che tra tutte le possibilita' offerte dai computer ad alte prestazioni, la soluzione di una rete di transputer si presenta come quella con il piu' alto rapporto prestazioni/costo, relativamente alla applicazione della ricostruzione 3-D di immagini ad ultrasuoni.

Dallo studio dei problemi concernenti la programmazione di una rete multiprocessore per la parallelizzazione dell'algoritmo e dal confronto delle prestazioni tra vari processori, si sono individuati i seguenti vantaggi della soluzione basata su

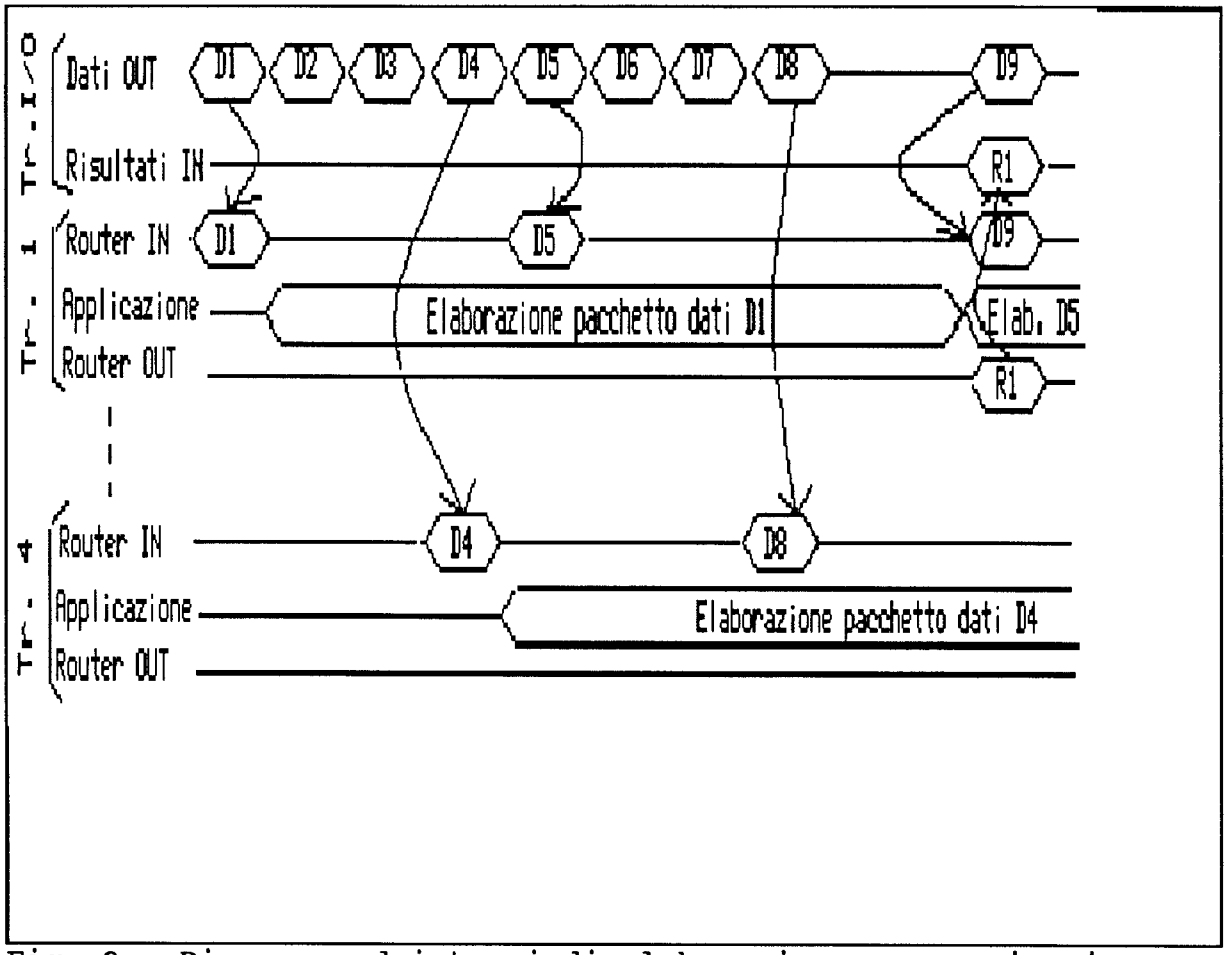


Fig. 9 - Diagramma dei tempi di elaborazione e comunicazione

transputer :

1) Prestazioni elevate del T800-XX valutate con programmi di benchmark orientati all'applicazione specifica

2) Tempi brevi di sviluppo del software per la gestione della rete di transputer in linguaggio OCCAM (Metodo "Message passing")

3) Efficienza, modularita' e semplicita' della programmazione in OCCAM (quindi affidabilita'del software)

4) Costi competitivi con altri processori ad alte prestazioni, considerando il grande supporto tecnico hard. & soft. fornito per lo sviluppo di applicazioni specifiche con il transputer.

Infine per una "transputer farm" sono state sviluppate alcune formule di progetto per il dimensionamento del sistema (memoria e numero di processori) e sono stati definiti i moduli software per

la gestione della rete.

Il lavoro futuro e' diretto alla verifica con ulteriori prove sperimentali della efficienza dei vari compilatori C oggi disponibili e lo studio di eventuali problemi di "collo di bottiglia" del sistema in presenza di molte richieste simultanee di accessi al disco.

Desisedero infine ringraziare il Prof. Martin Neumann, il Dott. Riccardo Pini, l'Ing. Federico Boragine, per la loro collaborazione nella raccolta di dati preziosi per questo report.