COMPILACIÓ PER A SUPERCOMPUTADORS

JORDI TORRES I EDUARD AYGUADÉ
Centre Europeu de Paral·lelisme de Barcelona
Departament d'Arquitectura de Computadors. Universitat Politècnica de Catalunya. Gran Capità, s/n. Mòdul D6, 08071 Barcelona

1. INTRODUCCIÓ

Estem a tocar de l'any 2000. Al nivell muncial ens trobem immersos en una cursa per la productivitat, la qualitat i la innovació. El software i les seves aplicacions, cada cop més complexes, demanen una capacitat i velocitat de càlcul cada vegada més importants. Una ciència que pretén donar resposta a aquestes necessitats és la que es coneix per SUPERCOMPUTACIÓ.

S'associa el terme de Supercomputador a aquells computadors que han estat dissenyats per a poder executar gran nombre d'operacions per segon, sobre nombres codificats normalment en coma flotant, i en cada època, són els que efectuen major nombre d'operacions per segon. Al principi dels anys 70, la velocitat punta d'aquestes màquines estava per sota dels 100 MFlops (10^8 operacions/segon). En l'actualitat, existeixen supercomputadors amb velocitats punta que ronden els GFlops (10^{11} operacions/segon). Aquesta velocitat sostinguda, junt amb la gran capacitat de les memòries i dels perifèrics d'entrada/sortida, permetrà de resoldre un conjunt d'aplicacions que fins ara han estat considerades com grans reptes per les administracions d'Europa, els Estats Units d'Amèrica i el Japó [GrCh92,Zane92].

La supercomputació no és una disciplina senzilla sinó ben al contrari, el resultat de la interrelació de diverses àrees de recerca: Aplicacions, Llenguatges d'Alt Nivell, Compiladors, Mètodes Numèrics, Sistemes Operatius, Arquitectura de Computadors i Disseny del hardware bàsic [VaLL92]. L'avanç en el camp de la Supercomputació és possible gràcies a una estreta col·laboració entre grups de recerca d'aquests temes.

Els supercomputadors han esdevingut una eina fonamental per a la recerca científica. Els computadors han estat durant anys, eina de suport i anàlisi per a la realització d'experiments científics, pero els supercomputadors poden fer molt més que això: fan també un paper primordial en el món científic, ja que permeten construir i verificar models en la recerca de teories que per exemple puguin descriure fenòmens naturals. Aquests complexos models matemàtics normalment requereixen

trilions d'operacions per a ésser resolts, i era impossible resoldre'ls en un temps raonable amb els computadors tradicionals.

D'entre l'ampli ventall d'àrees en què es pot aplicar la potència dels supercomputadors hi podem trobar: la biologia, la recerca energètica, l'astrofísica, la enginyeria aeroespacial, la prospecció petrolífera, el processat d'imatge, la indústria d'automoció, l'enginyeria genètica... En contret cal destacar les aplicacions que són considerades com els grans reptes per diferents administracions [GrCh92] que tenen programes específics orientats a situar-les en la posició de capdavanteres en el desenvolupament d'aquestes tecnologies. Les que més destaquen són: modelat del clima, turbulències de fluids, models globals, oceà-atmosfera, cromodinàmica quàntica, disseny de nous components químics, disseny de nous materials, disseny integrat d'avions i cotxes.

2. Arquitectura dels supercomputadors

Dos dels factors principals que han contribuït d'una manera fonamental en l'increment de la potència dels supercomputadors són: la *tecnologia* i la *concurrència.*

Els avenços de la tecnologia, tant en materials utilitzats en la construcció de dispositius de commutació, capacitat d'integració, així com en la capacitat d'encapsular els dispositius en circuits integrats, estan permetent de reduir el temps de càlcul de funcions extraordinàriament complexes. El dissenyador és qui, en funció de l'estat de la tecnologia, decideix què utilitzar per a obtenir la velocitat de càlcul desitjada junt amb altres paràmetres de disseny com són mida, consum, ...

Per concurrència s'entén la possibilitat d'efectuar diverses operacions simultàniament en un mateix instant. Les tècniques bàsiques s'anomenen *segmentació* i *paral·lelisme* [HoJe81].

La tècnica de segmentació consisteix a descompondre una determinada operació en N suboperacions a realitzar en fases o etapes diferents. D'aquesta manera, una operació es realitza a mesura que la informació involucrada en aquesta operació travessa les N etapes. En un mateix instant de temps tenim N operacions diferents en progrés (cadascuna d'elles en una fase diferent). Aquesta tècnica pot aplicar-se (a) al nivell d'execució d'instruccions (això dóna lloc als anomenats processadors escalars segmentats i supersegmentats) o (b) al nivell de operacions sobre dades (això dóna lloc als anomenats computadors vectorials segmentats).

D'altra banda, per paral·lelisme s'entén la realització concurrent de càlculs, iguals o diferents, sobre diferents conjunts de dades per part de diverses unitats funcionals. La tècnica de paral·lelisme també pot ésser aplicada (a) al nivell d'execució d'instruccions (donant lloc als processadors superescalars o amb diversos fluxos d'execució, computadors *Very Long Instruction Word* i sistemes multiprocessadors) o (b) al nivell d'operacions sobre dades (els anomenats processadors en *array*).

En l'actualitat, els supercomputadors més utilitzats es basen en una arquictectura de multiprocessador en què cada element de procés és un processador vectorial

segmentat. Això s'ha esdevingut en part pel bon ús de la tecnologia que permeten aquests sistemes i en part per la facilitat de detectar, explotar i expressar el paral·lelisme vectorial que ofereixen la major part de les aplicacions en enginyeria i ciència. Algun d'aquests sistemes són els supercomputadors Cray Y-MP o Convex C3. El nombre de processadors va de 2 a 8 i tots ells comparteixen una memòria dividida en mòduls per a augmentar l'amplada de banda del subsistema de memòria. En aquest tipus d'arquitectura la xarxa d'interconnexió és un factor que limita molt l'escalabilitat. Aquest és el motiu principal que dóna força als sistemes amb memòria distribuïda, on cada element de procés disposa de la seva pròpia memòria, cosa que permet una escalabilitat més elevada del sistema. La topologia de la xarxa d'interconnexió defineix com estan connectats entre si els elements de procés (lineal, malla, hipercub, ...). Alguns dels més coneguts són el Supernode de 16 processadors escalabre fins a 256 o l'Hipercub d'Intel amb 1024 processadors.

De tota manera els reptes fins a l'any 2000 necessiten una potència de càlcul difícilment aconseguible amb les implementacions actuals. Tots els fabricants estan abocats al disseny de sistemes anomenats *Massive Parallel Processors MPP.* En ells, la combinació jeràrquica de les fiolosofies de memòria compartida i distribuïda, juntament amb l'ús de processadors superescalars i un gran nombre d'elements de procés es preveu que donarà pas a la potència de càlcul que es necessita.

3. Interès dels supercompiladors

En general hi ha diverses alternatives per a programar un supercomputador per tal d'explotar-ne la seva poténcia. Totes esten en funció del grau de coneixement que el programador d'aplicacions té de l'arquitectura del seu supercomputador.

Una primera opció és la de programar l'aplicació usant un llenguatge amb construccions que permetin especificar la concurrència que existeix en l'aplicació. La tasca de detectar i saber explotar eficientment aquesta concurrència per no ésser gens fàcil i requerir un coneixement profund de l'arquitectura del supercomputador.

Una segona opció és la de programar l'aplicació usant un llenguatge seqüencial convencional i deixar al compilador la feina de generar el programa concurrent equivalent. L'us de llibreries amb un nucli bàsic de funcions ja programades (i per tant adaptades a l'arquitectura en qüestió) complementa aquesta opció.

Un dels principals problemes amb què ens trobem en disposar d'un supercomputador és el de convertir els codis de les aplicacions existents (milions de línies de programes ja escrites, els anomenats "*dusty deck programs*") per tal de poder-les executar de forma eficient sobre el nou supercomputador. A primera vista això significaria transformar manualment tots els programes amb la conseqüent pèrdua de temps i el risc d'errors.

Els SUPERCOMPILADORS, que fan una reestructuració automàtica del codi seqüencial, han estat desenvolupats, entre d'altres motius, per a poder solucionar aquest problema. Aquests compiladors analitzen programes seqüencials per tal de detectar i explotar de forma eficient la concurrència inherent al programa seqüencial.

La utilització dels reestructuradors automàtics va més enllà del simple fet de poder transformar els "*dusty deck programs*". Tot i l'existència de llenguatges concurrents sembla avantatjós continuar programant amb un llenguatge seqüèncial i transformar-lo després d'una forma automàtica en una versió concurrent. Dues de les raons més acceptades són les següents: d'una banda, la programació de les aplicacions en un llenguatge seqüencial independent de l'arquitectura permet la fàcil portabilitat entre diferents arquitectures, mentre que les versions concurrents normalment són força dependents de l'arquitectura del anell supercomputador. D'altra banda, és més difícil desenvolupar, depurar i mantenir aplicacions concurrents que les seves versions equivalents seqüencials a causa de la forta dependència amb l'arquitectura.

4. Entorns de programació paral·lela

En general, la complexitat de la relació entre l'estructura dels algorismes, l'arquitectura del supercomputador i les estratègies de transformació impossibilita una solució òptima global i ens obliga a utilitzar certes heurístiques. En aquest sentit són molts els factors que indiquen que una bona manera de tractar la reestructuració és mitjançant sistemes interactius, on l'usuari pot fer certes decisions i donar informació al sistema que no pot ésser obtinguda de manera automàtica a partir del programa seqüencial.

Els elements principals en un entorn de programació paral·lela (figura 1) acostumen a ésser: (a) llenguatges amb construccions específiques que permetin expressar el paral·lelisme de les aplicacions i els compiladors associats, (b) llibreries amb operacions bàsiques eficientment implementades, (c) eines

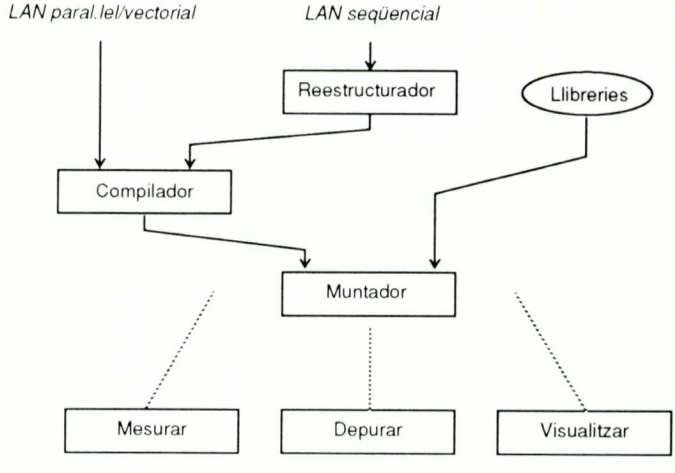

Fig. 1. Eines més comunes dins un entorn de programació paral·lela.

d'anàlisi i reestructuració de programes i (d) eines que permetin la depuració d'aplicacions paral·leles, la mesura de l'eficiencia en una determinada arquitectura i eines de visualització. A continuació es realitza una breu descripció de cadascuna d'aquestes eines.

4.1. *Llenguatges de programació*

És d'esperar que en un entorn de programació paral·lela s'ofereixin al programador d'aplicacions diferents tipus de llenguatges, incloent-hi tant llenguatges seqüencials com paral·lels.

L'alternativa més utilitzada en el desenvolupament de llenguatges paral·lels ha estat la d'estendre els llenguatges seqüencials convencionals existents incorporant-hi construccions i primitives orientades a especificar l'execució concurrent (vectorial i/o paral·lela) d'operacions. Aquestes extensions pretenen obtenir el màxim profit de les característiques d'aquestes arquitectures per als càlculs a què estan orientades.

Per exemple, per a programar un sistema multiprocessador amb memòria compartida (SMM) és necessari descompondre els càlculs a realitzar en un conjunt de tasques que puguin executar-se en paral·lel i de forma asíncrona. En aquests sistemes, les dades es troben en una memòria comuna i són accessibles per qualsevol dels processadors del sistema. Cada processador es cuida d'executar una o més d'aquestes tasques en funció d'una estratègia de planificació. A la figura 2 es veu el procés de programació d'aquest tipus de sistemes.

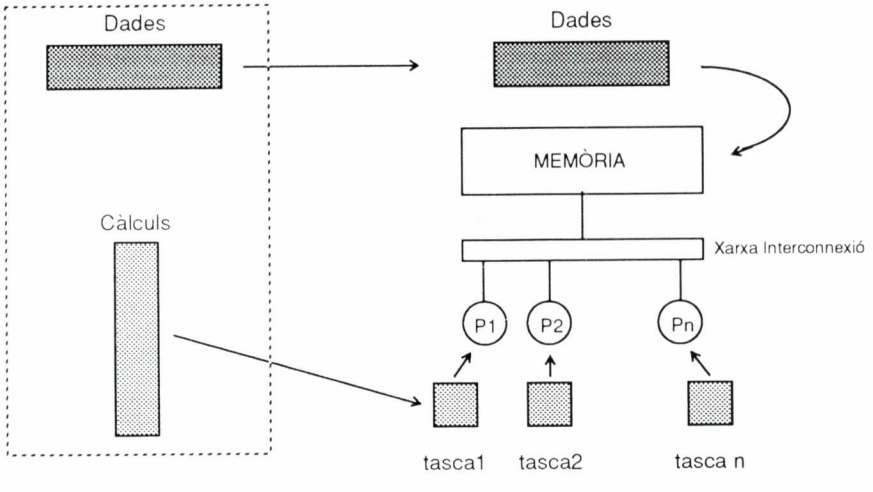

Fig. 2 Descomposició d'un problema en tasques per a la seva execució en un sistema multiprocessador amb memòria compartida.

Les construccions introduïdes als llenguatges seqüèncials permeten al programador d'aplicacions: (a) caracteritzar les variables utilitzades, (b) la creació i monitorització de tasques, (c) especificar bucles paral·lels i (d) sincronitzar de forma explícita l'execució de les tasques generades. Cal dir que el paral·lelisme potencial en les aplicacions científiques i d'enginyeria es troba en càlculs que es fan de forma repetida un gran nombre de vegades sobre una estructura de dades regular tipus matriu. Per això, les construccions dels llenguatges concurrents orientades a expressar la concurrència dels bucles tipus DO són les més importants. Les més utilitzades són DOALL i DOACROSS. En un bucle DOALL totes les iteraccions del bucle poden ésser executades en paral·lel i en qualsevol ordre. No obstant això, en un bucle DOACROSS no pot iniciar-se una determinada iteració del bucle fins que totes les anteriors hagin estat iniciades. Aquest últim s'utilitza quan existeixen dependències entre iteracions del bucle, cosa que facilita la inserció de primitives de sincronització.

En el cas d'un sistema multiprocessador amb memòria distribuïda (DMM) no és suficient descompondre els càlculs a realitzar, també és necessari distribuir les dades entre les memòries locals dels elements de procés. La figura 3 mostra el procés de programació d'aquests tipus de sistemes.

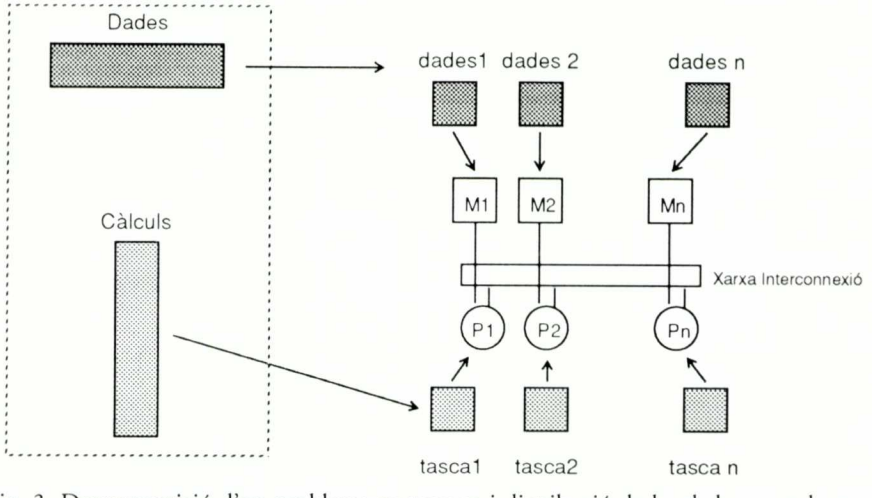

Fig. 3. Descomposició d'un problema en tasques i distribució de les dades per a la seva execució en un sistema multiprocessador amb memòria distribuïda.

Els llenguatges de programació seqüencial són estesos amb primitives que permeten, a més, l'espécificació de la distribució de dades a realitzar. Així per exemple el Fortran D [FHKK92], el High Performance Fortran [Digi92] i el Vienna Fortran [ZBCM92], entre d'altres, inclouen primitives que permeten.

- definir la topologia d'interconnexió del sistema,
- descompondre una matriu en blocs rectangulars,
- definir quin processador emmagatzema en la seva memòria local un determinat bloc.

També permeten especificar la comunicació entre elements de procés, tant al nivell de comunicació global (tipus BROADCAST) com entre parelles d'element de procés (tipus SEND i RECEIVE).

La majoria de les vegades la programació d'aplicacions paral·leles requereix un coneixement del hardware sobre el qual s'executarà l'aplicació. Per això, la portabilitat és una de les característiques que es fan difícils en la fase de mantenimient d'una aplicació quan el programador explicita el paral·lelisme.

En conclusió, si es tenen en consideració aquests aspectes, la programació en un llenguatge seqüencial continua tenint vigència i és una bona alternativa per a programadors no experimentats en la supercomputació.

4.2. *Llibreries bàsiques*

El programador ha de disposar d'un nucli bàsic de rutines programades de forma eficient pel supercomputador que executarà les aplicacions. Normalment són una bona ajuda de claredat, portabilitat, modularitat i manteniment dels codis, ja que amb el temps han esdevingut força estandarditzades. Existeixen nombroses llibreries especialitzades en diferents àmbits. Moltes d'elles, com poden ésser per exemple un conjunt de rutines bàsiques d'àlgebra lineal (LINPACK [DBMS79], BLAS [LHKK79]. ...), estan en l'actualitat programades per a un ampli ventall de computadors. Aquestes rutines alhora han esdevingut elements de comparació de l'eficiència de diferents arquitectures.

4.3. *Eines d'anàlisi i reestructuració*

Com ja s'ha vist, una de les alternatives actuals a la programació paral·lela consisteix en la codificació d'aplicacions en un llenguatge seqüencial convencional, deixant al compilador la tasca de detectar i explotar la concurrència inherent del programa. En la figura 4 es mostra en línies generals aquest procés de reestructuració.

En general són tres las fases en què pot considerar-se dividit aquest procés de reestructuració:

- *detecció i normalització de bucles:* en aquesta fase s'apliquen diferents tècniques convencionals dins el camp de la compilació i altres de més específiques orientades a facilitar el posterior procés d'anàlisi de dependències i optimitzacions de codi. Dins aquesta fase també es poden fer mesures estàtiques de característiques dels bucles (com per exemple el paral·lelisme i la longitud vectorial màxima [ALTL90]).

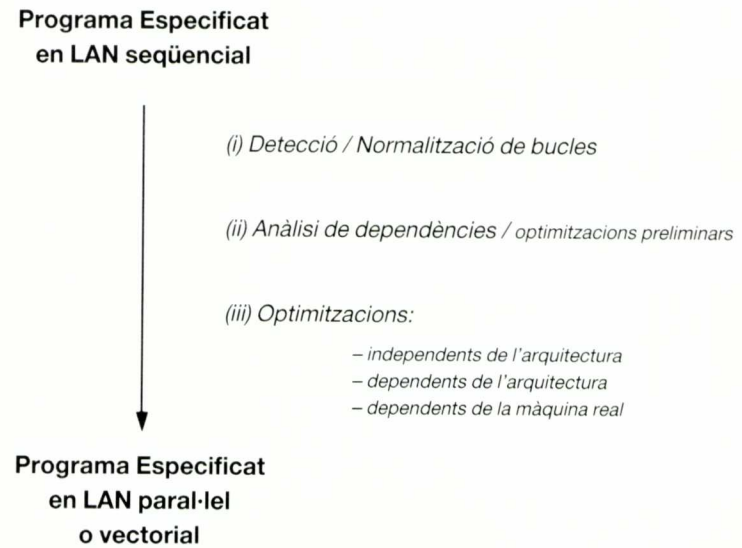

Fig. 4. Fases en el procés de reestructuració de bucles.

- *anàlisi de dependències entre sentències del bucle:* en aquesta fase es determinen les relacions d'ordre entre operacions que es realitzen en el bucle i que han de preservar-se si es desitja mantenir la semàntica del programa seqüencial original.
- *optimitzacions:* aplicació de tècniques específiques amb la finalitat d'extreure el paral·lelisme inherent de l'aplicació i la seva expressió en forma d'un programa especificat en un llenguatge paral·lel o vectorial [KKPL81,PaWo86]. Aquestes transformacions poden ésser independentes de l'arquitectura, dependents de l'arquitectura o específiques del supercomputador sobre el que s'ha d'executar l'aplicació.

El procés de reestructuració per a sistemes amb memòria distribuïda comporta, a més de generar tasques paral·leles, l'optimització de la ubicació de les dades en les memòries locals dels elements de procés. La penalització que s'ha de pagar pel fet d'accedir a informació no local és molt gran, i és molt important la bona ubicació de les dades.

En aquests darrers anys han aparegut algunes eines d'aquestes, tant per a l'àmbit universitari (Parafrase II en la Univeristy of Illinois [KKLW80,PGHL89] o PFC i ParaScope en la University of Rice [AlKe87,BKKM89]) com per a l'àmbit comercial (supercompiladors dels supercomputadors actuals com per exemple KAP de Kuck and Associates [Kuck88] o VAST de Pacific Sierra).

La interacció d'aquestes eines amb l'usuari adquireix una elevada importància ja que pot guiar i donar informació addicional a l'eina de reestructuració. Així, per exemple, l'usuari pot forçar la vectorització o la

paral·lelització en situacions en què el compilador no disposa de tota la informació que necessita per a assegurar la validessa semàntica de la reestructuració.

En el CEPBA s'està treballant en un entorn anomenat GTS per a la paral·lelització automàtica de bucles tant per multiprocessadors amb memòria compartida [Aygu89,ATLL91] com per multiprocessadors amb memòria distribuïda [TALL91].

4.4. *Eines de depuració, mesura i visualització*

La depuració d'una aplicació paral·lela és una fase important en tot el procés de programació. Una aplicació semànticament pot ésser correcta però això no vol dir que sigui eficient en ésser executada per un supercomputador d'unes determinades característiques. Les eines de mesura proporcionen informació sobre l'aplicació, ja sigui monitoritzant la seva execució en un simulador o en la màquina real.

Les mesures sobre un programa seqüencial permeten al programador identificar aquelles parts del programa que s'executen amb més freqüència o consumeixen la major part del temps d'execució (per sentència, bucle o procediment) o la freqüència amb què són agafades certes branques en sentències condicionals. Amb mesures similars sobre la versió paral·lela, és possible identificar aquelles parts que requereixen un major esforç en les fases anteriors del cicle de realització de l'aplicació.

Els entorns de programació paral·lela estan suportats normalment per eines gràfiques que permeten visualitzar tota la informació que l'usuari necessita per a portar a terme el seu treball (visualització del codi seqüencial, dependències, el resultat de l'aplicació d'una determinada tècnica de reestructuració...) [BKKM89]. D'altra banda, es requereixen sofisticades eines de visualització i gràfics si es desitja obtenir el màxim profit del volum d'informació que pot obtenir-se monitoritzant l'execució d'una aplicació paral·lela.

5. El futur dels supercompiladors

Tot i l'elevat nombre d'estudis al voltant d'aquest tema, els resultats actuals no són els desitjats. Els supercompiladors actuals no són capaços de transformar tots els programes de forma eficient. La qualitat del resultat depèn totalment de l'estructura de l'algorisme i de l'arquitectura de la màquina. Mentre que en arquitectures vectorials multiprocesadores (amb memòria compartida) o arquitectures en *array* ja s'ha aconseguit dissenyar compiladors que generin un codi acceptablement eficient en força casos, en els *Massive*

Parallel Processors no passa el mateix. I precisament l'estat inicial dels compiladors per a aquestes màquines és una de les restriccions més fortes en l'actualitat per a facilitar el desenvolupament i l'ús d'aquestes potents arquitectures.

En [EHL91] es presenten els resultats obtinguts pel supercompilador de l'Alliant FX/80 en una sèrie de programes representatius de les necessitats científiques del moment. En general el codi paral·lel és poc més ràpid que la seva versió seqüencial, en algun cas fins i tot és pitjor. El problema normalment radica en el fet que les diverses tècniques de reestructuració automàtica que coneixem actualment només són eficients per a un conjunt reduït de problemes. Ara bé, hi ha estudis [EHLP91,EiBl91] que demostren que es poden obtenir més bons resultats en el futur, aprofundint en l'anàlisi dels codis. De fet, amb molt poques modificacions, les tècniques actuals es poden estendre a més casos.

Resum

Vist l'auge que està prenent la computació com a eina bàsica de treball en temes de recerca científica, un dels principals reptes tècnics que ens trobem avui és el d'aprendre a aprofitar la gran capacitat de càlcul que ens ofereixen els actuals supercomputadors.

En aquesta línia van el que es coneix per eines de suport al desenvolupament de software per a supercomputadors. L'objectiu principal d'aquestes eines és tant el de donar assistència en el procés de formulació d'algorismes paral·lels, com el de mecanitzar el procés de traducció dels algorismes concebuts pels científics en una eficient implementació sobre l'arquitectura específica del supercomputador utilitzat.

En aquest article es presenten breument les arquitectures que donen cos als supercomputadors, i es fa un repàs de l'estat actual de les eines de suport al desenvolupament de software per a supercomputadors.

REFERÈNCIES

[AlKe87] J.R. ALLEN and K. KENNEDY, "Automatic Translation of FORTRAN Programs to Vector Form", ACM Trans. on Programming Languages and Systems, Vol. 9, núm. 4, octubre, 1987.

[ALTL90] E. AYGUADÉ, J. LABARTA, J. TORRES, J.M. LLABERIA and M. VALERO, "Parallelism Evaluation and Partitioning of Nested Loops for Shared Memory Multiprocessors", 3rd Workshop on Programming Languages and Compilers for Parallel Computing, Irvine California USA, agost, 1990.

[ATLL91] E. AYGUADÉ, J. TORRES, J. LABARTA, J.M. LLABERIA and M. VALERO, "Balanced Loop Partitioning using GTS", Languages and Compilers for Parallel Computing, Springer-Verlag, 1991.

[Aygu89] E. AYGUADÉ, "Paralelización Automática de Recurrencias en Programas Secuenciales Numéricos', Tesis Doctoral, Dep. d'Arquitectura de Computadors, Univ. Politècnica de Catalunya, octubre, 1989.

[BKKM89] V. BALASUNDARAM, K. KENNEDY, U. KREMER, K. MCKINLEY and J. SUBHLOK, "The ParaScope Editor: An Interactive Parallel Programming Tool", Proc. of the Supercomputing'89, Reno-Nevada, novembre, 1989.

[DBMS79] J.J. DONGARRA, J.R. BUNCH, C.B. MOLER and G.W. STEWART, "LINPACK Users'Gride" SIAM, Philadelphia, Pa., 1979.

[Digi92] Digital Equipment Corporation, Massively Parallel Systems Group, "High Performance Fortran: Proposal", gener, 1992.

[EiBl91] R. EIGENMANN and E. BLUME, "An Effectiveness Study of Parallelizing Compiler Techniques", Center for Supercomputing Research and Development, Univ. of Illinois at Urbana-Champaign, CSRD report no. 1090, maig, 1991.

[EHLP91] R. EIGENMANN, J. HOEFLINGER, Z. LI and D. PADUA, "Experience in the Automatic Parallelization of Four Perfect-Benchmark Programs", Center for Supercomputing Research and Development, Univ. of Illiniois at Urbana-Champaign, CSRD report no. 1193, agost, 1991.

[FHKK92] G. FOX, S. HIRANANDANI, K. KENNEDY, C. KOELBEL, U. KREMER, C.W. TSENG and M.Y. WU, "Fortran D Language Specification",

Rice University, Department of Computer Science, Report COMP TR90-141, gener, 1992.

[GrCh92] "Grand Challenges: High Performance Computing and Communications, The FY1992 U.S. Research and Development Program", Committee on Physical, Mathematical and Engineering Science U.S. 1992.

[HoJe81] R.W. HOCKNEY and C.R. JESSHOPE, "Parallel Computers Architecture, Programming and Algorithms", Adam Hilger Ltd, Bristol, 1981.

[KKPL81] D.J. KUCK, R.H. KUHN, D.A. PADUA, B. LEASURE and M. WOLFE, "Dependence Graphs and Compiler Optimizations", Proc. of the 8th ACM Symposium on Principles of Programming Languages, gener, 1981.

[§uck88] KUCK & Associates Inc., "KAP User's Guide", Champaign-Illinois, 1988.

[LHKK79] C.L. LAWSON, R.J. HANSON, D.R. KINCAID and F.T. KROHG, "Basic linear algebra subprograms for Fortran usage". ACM Trans. Math. Softw. 3, 3, pp. 308-323, 1979.

[PaWo86] D.A. PADUA i M. WOLFE, "Advanced Compiler Optimizations for Supercomputers", Communications of the ACM, vol. 29, n. 12, desembre, 1986.

[PGHL89] C.D. POLYCHRONOPOULOS, M. GIRKAR, M.R. HAGHIGHAT, C.L. LEE, B. LEUNG, D.SCHOUTEN, "Parafrase-2: An Environment for Parallelizing, Partitioning, Synchronizing and Scheduling Programs on Multiprocessors", Proc of 1989 Int'l Conf. on Parallel Processing, vol II, agost, 1989.

[TALL91] J. TORRES, E. AYGUADÉ, J. LABARTA, J.M. LLABERIA and M. VALERO, "On Automatic Loop Data-Mapping for Distributed-Memory Multiprocessors", Proceedings of The Second European Distributed Memory Computing Conference, München-Alemanya, abril, 1991.

[VaLL92] M. VALERO, J.M. LLABERIA, J. LABARTA, "Supercomputación: Una ciencia interdisciplinar", Jornades de Supercomputació a Catalunya, 27 i 28 maig, 1992.

[ZBCM92] H. ZIMA, P. BREZANY, B. CHAPMAN, P. MEHROTRA and A. SCHWALD, "Vienna Fortran- A Language Specification: Version 1.1", ICASE Internal Report 21, ICASE, Hampton, VA, 1992.

[Zane92] P. ZANE, "HPCC IN EUROPE", Highly Parallel Computing Systems seminar, 1992 IBM EUROPE INSTITUTE Oberlech, Àustria, juliol, 1992.