PROGETTO FINALIZZATO
SISTEMI INFORMATICI E CALCOLO PARALLELO

SOTTOPROGETTO 8

Iniziativa di Supporto al Calcolo Parallelo

Coordinatore Stefano Trumpy

R. Baraglia[1], G. Bartoli[1], D. Laforenza[1],
A. Mei[1], A. Laganà[2]

TRAJE: un programma per lo studio di sistemi gassosi atomo-diatomo sull'elaboratore nCUBE 2

N. 8/R48 Settembre 1993

[1] CNUCE, Italian National Research Council, Pisa, Italy
[2] Dept. of Chemistry, University of Perugia, Italy

Rapporto Tecnico

CNUCE - CNR, via S. Maria 36, 56126, Pisa, Italy

INDICE

1. Introduzione

Nel presente rapporto viene descritto il programma Traje implementato su nCUBE 2 usato per lo studio quasiclassico delle reazioni atomo-diatomo. Dopo la descrizione del metodo di calcolo quasiclassico, vengono descritte le scelte adottate per l'implementazione della versione parallela del programma Traje sull'elaboratore parallelo a memoria distribuita nCUBE 2. Successivamente viene descritto il comando che deve essere usato per eseguire il programma e il contenuto dei file di input e di output utilizzati.

2. Il metodo di calcolo quasiclassico

Il problema scientifico trattato riguarda il calcolo degli osservabili sperimentali delle reazioni atomo-diatomo elementari. La soluzione del problema porta alla determinazione di grandezze fisiche utile alla modellizzazione accurata dei sistemi gassosi complessi non in equilibrio. L'interesse per questi sistemi è motivato dalla importanza che essi rivestono nella realizzazione di molte delle moderne applicazioni tecnologiche (es. laser e plasmi freddi) [1].
La grandezza di interesse è, nel nostro caso, la sezione d'urto $\sigma_{vjv'j'}(E_{tr})$ (v e j sono rispettivamente lo stato vibrazionale e rotazionale dei reagenti; v' e j' indicano gli stessi stati per prodotti; E_{tr} è l'energia di collisione del sistema). Essa è definita come un integrale pentadimensionale (rispettivamente sulla orientazione iniziale θ del diatomo, sulla orientazione ω del momento angolare rotazionale del diatomo, sulla distanza b del vettore velocità dal centro di massa, sulla fase η dell'oscillazione del diatomo, sull'angolo β formato dal vettore velocità con il piano iniziale del triatomo a valori fissi di v, j e E_{tr}).

$$\sigma_{vjv'j'}(E_{tr}) = \int_{b=0}^{b_{max}} b \int_{\theta=0}^{\pi} \sin\theta \int_{\omega=-\pi}^{\pi} \int_{\eta=-\pi}^{\pi} \int_{\beta=0}^{2\pi} f(\theta,\omega,\eta,\beta,b,E_{tr}) d\theta d\omega d\eta d\beta db$$

La f è una funzione booleana pari ad uno quando, integrando le equazioni di Hamilton a partire dal sistema nello stato iniziale vj, si ottengono prodotti sullo stato finale $v'j'$ e zero altrimenti.
L'integrale pentadimensionale viene valutato numericamente usando il metodo Monte

Carlo. Il valore dell'integrale, cioè, viene approssimato dalla somma dei valori della $f_{vjv'j'}(\theta,\omega,\eta,\beta,b,E_{tr})$ per un numero N, sufficientemente grande, di eventi, statisticamente rappresentativi del processo; per ogni evento, la curva di inviluppo delle posizioni istantanee delle particelle viene denominata traiettoria.

I metodi Monte Carlo sono ampiamente usati per il calcolo di integrali multidimensionali grazie alla loro proprietà di essere particolarmente adatti alla elaborazione automatica. In generale, il loro obiettivo è approssimare il valore di un integrale definito:

$$Ig = \int_{I^S} g(t)dt \quad \text{mediante} \quad \frac{1}{N}\sum_{i=1}^{N} g(x_i)$$

dove I^S è il cubo unitario in s dimensioni, g una funzione continua in s variabili, x_i punti appartenenti al dominio di integrazione, scelti casualmente.

La qualità dell'approssimazione dipende dal valore del numero N di punti considerati e dalla capacità del generatore di numeri casuali di dare una stringa omogeneamente distribuita di valori nell'intervallo 0-1.

3. Struttura del programma Monte Carlo per il calcolo delle traiettorie

Nell'applicazione di dinamica molecolare in esame, la procedura per il calcolo sequenziale può essere schematizzata come mostrato in figura 1.

Tra i dati di input, oltre alle costanti fisiche necessarie per determinare altri parametri di utilità generale, abbiamo l'iniziatore della sequenza di numeri casuali e il numero N di traiettorie da calcolare. La sezione maggiore del programma* è comunque incorporata in un ciclo che opera sull'indice della traiettoria; una volta generati numeri pseudo-casuali che ne descrivono le condizioni iniziali, la parte più intensa dal punto di vista computazionale è l'integrazione della traiettoria. Per una analisi statistica significativa, è necessario disporre di un numero sufficientemente grande di eventi (in genere dell'ordine di 10^3, 10^4), la cui singola elaborazione richiede una quantità di tempo variabile e non prevedibile a priori, in quanto dipendente da una complessa serie di

* Il programma è stato derivato dal codice 273 del QCPE, Bloominghton, Indiana.

fattori. Tutto ciò si traduce in una domanda di tempo di calcolo che un elaboratore sequenziale difficilmente potrebbe soddisfare. Quindi, si è reso necessario portare l'applicazione in ambiente parallelo, considerato anche che il problema essenzialmente consiste di un insieme di calcoli indipendenti.

```
Input di dati iniziali
Valutazione di costanti di uso generale

loop_on_traject(1,N)
for ntraject=1 to N do
  generazione pseudo-random delle condizioni iniziali
    della traiettoria ntraject
  integrazione traiettoria ntraject
  aggiornamento risultati parziali
next ntraject

Output dei risultati
```

Fig. 1: Struttura del programma sequenziale per il calcolo delle traiettorie

4. Esplicitazione del parallelismo nel calcolo delle traiettorie

La natura fortemente asincrona del problema suggerisce di organizzare il calcolo complessivo come un insieme di processi indipendenti (worker), ciascuno dedicato all'integrazione di un sottoinsieme di traiettorie [2] [3] [4]. I punti di sincronizzazione tra i vari processi sono l'acquisizione dei dati iniziali, comuni a tutte le traiettorie, dei dati relativi alla specifica traiettoria e la fase finale di raccolta e analisi dei risultati parziali. Tali fasi, a seconda del modello di programmazione scelto, possono essere incorporate nella elaborazione dei worker o delegate ad un processo gestore (master).
Nella fase di esplicitazione del parallelismo, devono essere comunque attentamente considerati due aspetti:

(1) *garantire la riproducibilità delle esecuzioni parallele* in presenza di una sequenza di numeri pseudo-casuali, necessari per definire le condizioni iniziali della traiettoria.

La generazione di tale sequenza si basa su successive chiamate alla subroutine *random* che, dato un intero (*seed*), restituisce un nuovo intero (*new_seed*) e un numero pseudo-casuale (*rand*). In un codice scalare, l'elaborazione di traiettorie differenti è sequenziale e perciò la sequenza di chiamate alla subroutine random è univocamente determinata. Invece, in un ambiente parallelo, la sequenza di numeri pseudo-casuale viene ad essere nondeterministica, se non si adottano particolari accorgimenti. La riproducibilità è ottenuta imponendo una corrispondenza biunivoca tra l'indice i della traiettoria e il relativo insieme di condizioni iniziali I(i).

La tecnica adottata prevede inizialmente la generazione sequenziale di un insieme di seed {s1[i]:i=1,..,N}, in cui s1[i] è il primo seed della traiettoria i-esima. Successivamente ogni worker, a partire da tale insieme, per ogni traiettoria i di sua competenza, estrae il relativo primo seed s1[i] e provvede a generare gli insiemi S2[i] e I(i). Quest'ultimo insieme fornisce le condizioni iniziali della traiettoria i-esima. Pur venendo ad essere costituiti indipendentemente su ogni worker, ad ogni computazione I(1),..,I(N) saranno sempre gli stessi.

(2) *scegliere un metodo accurato per distribuire il carico di lavoro (workload)* ai vari worker, perché il tempo di elaborazione per una singola traiettoria è variabile. I metodi possibili sono i seguenti [5]:

- modello geometrico con divisione statica del lavoro. Ogni worker, all'inizio della sua elaborazione, seleziona, in base al suo numero progressivo, un sottoinsieme di traiettorie da elaborare. L'implementazione del modello geometrico richiede solo lievi modifiche al programma sequenziale, data la semplicità dell'approccio. Come contropartita si può verificare una non uniforme distribuzione del carico computazionale tra i nodi usati (condizione di sbilanciamento del carico).
- modello task-farm con divisione dinamica del lavoro. In questo caso è necessario strutturare il codice secondo due componenti: un processo farmer che distribuisce pacchetti di lavoro (batch di traiettorie) ai processi worker . Un nuovo pacchetto di lavoro viene inviato dal master ai worker ogniqualvolta essi hanno completato il pacchetto di lavoro precedentemente assegnato. Una ulteriore scelta riguarda il numero di traiettorie costituenti un pacchetto di lavoro (granularità); assegnare una traiettoria per volta, invece che più traiettorie, porta ad un miglior bilanciamento del carico, ma si ha un maggior overhead di comunicazione. Aumentando la granularità si ha una diminuzione del bilanciamento del carico perché, non essendo stimabile a

priori la durata del calcolo di una singola traiettoria, si possono avere pacchetti di lavoro la cui elaborazione richiede tempi di esecuzione differenti. Pertanto, alla terminazione dell'esecuzione dell'ultimo pacchetto di lavoro, i nodi che hanno completato il calcolo del loro batch di traiettorie debbono rimanere inattivi in attesa del completamento dell'elaborazione su nodi computazionalmete più carichi.

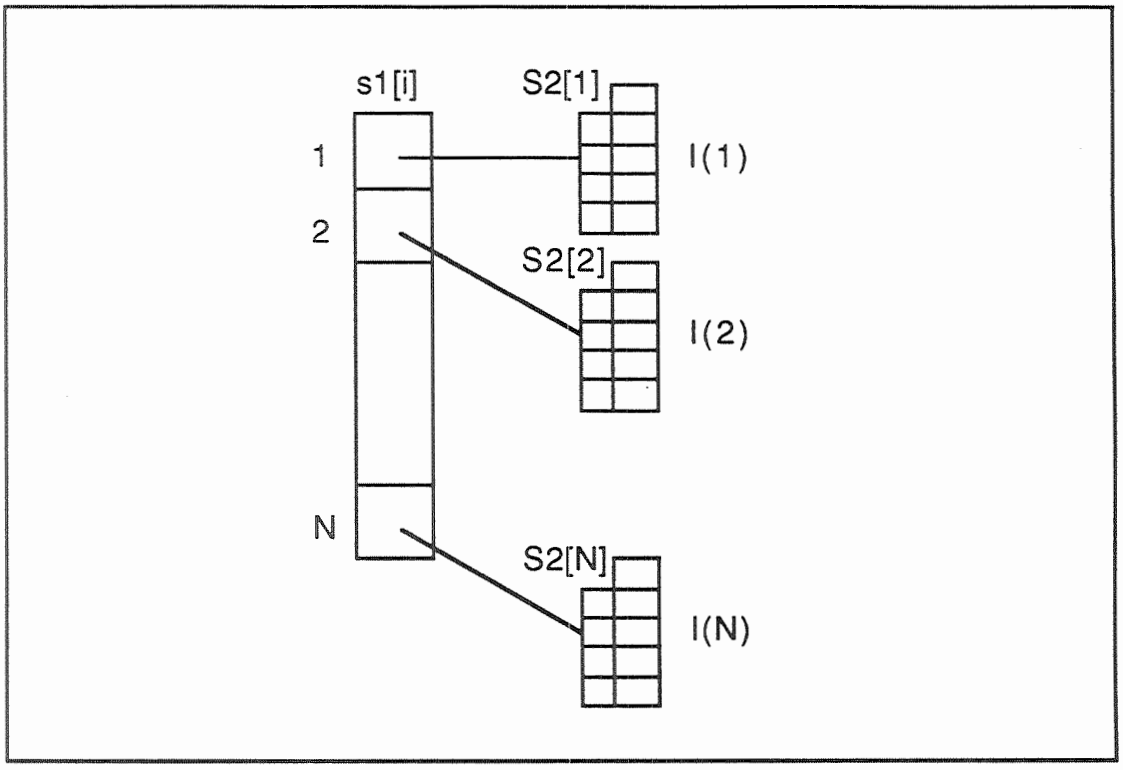

Fig. 5: Seed (s1[i] e S2[i]) e condizioni iniziali I(i) delle traiettorie: suddivisione del calcolo per la riproducibilità delle esecuzioni

4. Ristrutturazione parallela del programma sequenziale per l'elaboratore nCUBE 2

In lavori precedenti il codice sequenziale dell'applicazione è stato ristrutturato al fine di esplicitarne il parallelismo in ambienti MIMD-SM e MIMD-DM, utilizzando, rispettivamente, un multiprocessore IBM 3090 (mod. J-600)[6][7] e un ipercubo nCUBE (mod.4 con 16 nodi)[8][9]. In ambedue i casi la versione parallela del programma è stata ottenuta usando il modello di parallelismo task-farm.

Nel secondo caso, la versione parallela ottenuta prevedeva un processo farmer in esecuzione su Personal Computer costituente il computer host dell'ipercubo e un processo worker replicato su tutti i nodi dell'ipercubo. Una più efficiente realizzazione è comunque un task farm, strutturato come un unico codice, secondo lo stile di programmazione SPMD (Single Program Multiple Data)[6][10]. Ciò è stato reso possibile dalla adeguata disponibilità di memoria locale sui nodi dell'ipercubo attualmente in dotazione all'Istituto CNUCE di Pisa; 4 MByte sono stati sufficienti infatti per: memorizzare il programma, allocare le strutture dati, i buffer per la memorizzazione dei messaggi e le aree usate per soddisfare richieste di allocazione dinamica di memoria da parte del sistema operativo di nodo. Inoltre, questo stile di programmazione permette di sfruttare le funzionalità globali offerte dall'ambiente di programmazione nCUBE [10][11]. Ad esempio, la modalità globale di I/O prevede che tutti i nodi richiedano la stessa operazione di I/O che verrà effettivamente eseguita dal solo processore 0. Nel caso di operazione di input i dati letti sono trasferiti dal nodo zero a tutti gli altri nodi, nel caso di operazione di scrittura è il solo nodo 0 ad inviare i dati sui dispositivi di output.
Come mostrato in figura 3, nel modello task farm così implementato, il processo master va in esecuzione sul nodo 0, mentre il processo worker è replicato sui nodi (1,..,p-1), essendo p il numero di nodi allocati. Da notare che alcune parti di codice sono comuni: input, inizializzazione, calcolo dei seed, elaborazione finale e output.

Nel modello task farm, il nodo master rappresenta un punto critico di centralizzazione; una implementazione inefficiente andrebbe a degradare le prestazioni dei nodi worker e quindi la performance dell'applicazione. Una tale degradazione potrebbe verificarsi soprattutto se è usato un alto grado di parallelismo (consistente numero di processi worker). Al fine di aumentare l'ampiezza di banda del farmer, si è cercato di semplificare il programma e ridurre il numero di byte trasmessi in ogni messaggio. Ciò è stato ottenuto, principalmente, replicando la generazione dell'insieme dei seed (s(i):i=1,..,N) su ogni worker, in modo da evitarne una gestione centralizzata a carico del farmer. Conseguentemente è stato possibile usare solo due valori interi, delimitanti il sottoinsieme di traiettorie da integrare, nel messaggio di invio di un nuovo pacchetto di lavoro. I risultati intermedi della computazione sono accumulati sui nodi worker i quali, finita l'elaborazione di un batch di traiettorie, inviano al farmer un semplice

messaggio di sincronizzazione per segnalare la loro disponibilità a ricevere un nuovo pacchetto di lavoro. La collezione dei risultati intermedi avviene al termine del calcolo di tutte le traiettorie utilizzando funzioni globali di somma[10][11].

```
Lettura dei dati di input (numero di traiettorie da
  calcolare, condizioni iniziali globali, ecc.)
Calcolo delle condizioni globali
Costruzione dell'insieme dei seed usati per il calcolo delle
  singole traiettorie

Assegnazione del lavoro (batch di traiettorie) ai worker da
  parte del farmer
(nodo 0):
     if (# identificatore nodo .eq. 0) then

       Selezione di un batch di traiettorie in base alla
         granularità specificata
       Invio a ciascun worker il primo batch di traiettorie

       while (fino a che esistono bacth di traiettorie)
            Attesa e ricezione da ogni worker del messaggio
              indicante
            il completamento dell'elaborazione di un batch di
              traiettorie
            Invio al worker di un ulteriore batch di
              traiettorie
       end_while

         Invio di un messaggio di fine lavoro a ciascun
           worker

                              else
       while (fino a che esistono bacth di traiettorie)
       Ricezione del batch di traiettorie dal farmer
       for ciascuna traiettoria nel batch
            selezione del corrispondente seed
            generazione delle condizioni iniziali
            integrazione della traiettoria
            aggiornamento degli indici statistici
       end_for
       Invio al farmer del messaggio di fine elaborazione del
         batch
       end_while
end_if
call global routine per la collezione dei risultati
Esecuzione dell'analisi statistica
Registrazione dei risultati nel file di output
```

Fig. 3: Struttura della versione parallela del programma per il sistema nCUBE 2

In tabella 1 sono riportati i valori di alcuni indici di prestazione [11] ottenuti eseguendo sull'elaboratore nCUBE 2 il programma Traje al variare della granularità. Come si può vedere le migliori prestazioni si ottengono utilizzando granularità uguale ad 1. Quest'ultima è stata adottata nell'implementazione della attuale versione parallela del programma.

Tabella 1: Tempo di esecuzione (in secondi), speedup e efficienza per elaborazioni di 4096 traiettorie su 128 nodi

Granularità	1	2	4	8	16
Tempo di esec	435,5	441,8	459,5	481,8	588,8
Speedup	119,2	117,1	113,3	107,4	89,5
Efficienza	0,93	0,91	0,88	0,83	0,69

4. Esecuzione e descrizione dei file di input e di output

Il programma Traje può essere eseguito da qualsiasi *userid* presente sulla workstation "suncube" che è l'elaboratore host di nCUBE 2. Traje è scritto nel linguaggio di programmazione Fortran e usa le unità standard per le operazioni di input (unità 5) e output (unità 6). Pertanto i file di input e di output associati al programma in ciascuna esecuzione possono essere specificati utilizzando la modalità di ridirezione delle operazioni di I/O usata dal sistema operativo Unix.
Per attivare l'esecuzione del programma deve essere eseguito il seguente comando:

run -d*n* traje <*file.input* >*file.output*

in cui *n* è la dimensione dell'ipercubo richiesto che può variare tra il valore 0 e il valore 7. Va da se che se i file di input e di output non fanno parte della sottodirectory in cui si è posizionati al momento dell'esecuzione del comando *run* la specifica di detti file deve contenere il path necessario per la loro identificazione nel file system. Ad esempio il comando:

run -d *4* ntraje < *pippo/traiettorie/file1.input* > *pippo/output/file1.output*

utilizza un ipercubo di dimensione 4 (16 nodi), il file di input *file1.input* allocato nella sottodirectory *traiettorie* e il file di output *file1.output* allocato nella sottodirectory *output*. Ambedue le sottodirectory appartengono alla userid *pippo*.

In tabella 2 è mostrato un esempio di file di input del programma traiettorie. In esso i differenti tipi di record sono indicati con i valori (1), (2), ecc. che hanno il significato seguente:

(1) masse dei tre atomi A, B e C collidenti;

(2) costanti spettroscopiche (posizione di equilibrio r_0 e costante di forza γ per le tre coppie biatomiche AB, BC e AC);

(3) numero di termini del potenziale (l grado dello stesso. Infatti il tipo 4 è un potenziale espresso come polinomio delle variabili bond order definite come: $V = \sum_i \sum_j \sum_l a_{ijl} \mu_1^i \mu_2^j \mu_3^l$ con $\mu_i = \exp[-\gamma_i(r_0 - r_{0i})]$);

(4) termini del polinomio (coefficiente e potenze delle tre variabili bond order μ_1, μ_2 e μ_3);

(5) tipo del potenziale e variabile per stampare le mappe del potenziale;

(6) opzione per la generazione delle condizioni iniziali;

(7) numero di traiettorie, numero massimo dei passi di integrazione, numero di cicli per la frequenza della stampa dei valori intermedi e passo di integrazione;

(8) distanza massima raggiungibile;

(9) indicatore per la selezione delle condizioni iniziali e valore della condizione per:
   a) l'energia traslazionale (E_{tr}) o la velocità iniziale;
   b) l'angolo β;
   c) il parametro d'urto b;
   d) l'angolo θ;
   e) lo stato rotazionale j (o la temperatura rotazionale T_{rot});
   f) la fase del vibratore η;
   g) l'orientazione ω del vettore momento angolare di rotazione;

(10) distanza iniziale A - BC ;

(11) iniziatore della sequenza dei numeri pseudo-random.

Tabella 2: Esempio di file di input del programma Traje

```
(1)     6.939          18.9984          1.008
(2)     1.56386     0.97093
        0.916808    2.19424
        1.59570     1.17084
(3)     77 5
(4)     -0.3170699E+03   1   0   0
         0.3167180E+03   2   0   0
        -0.2325464E+03   3   0   0
         0.9531825E+02   4   0   0
        -0.2934111E+03   0   1   0
         0.1774375E+03   0   2   0
        -0.3940151E+02   0   3   0
         0.1418519E+02   0   4   0
        -0.1247705E+03   0   0   1
         0.8016479E+02   0   0   2
        -0.1801797E+02   0   0   3
         0.4623716E+01   0   0   4
         0.9224052E+03   1   1   0
         0.3272860E+03   1   0   1
         0.3381706E+03   0   1   1
        -0.1048643E+04   2   1   0
        -0.2962569E+03   1   2   0
        -0.3839140E+03   2   0   1
        -0.7478068E+03   1   1   1
        -0.7533124E+03   0   2   1
        -0.1715470E+03   1   0   2
        -0.2134407E+01   0   1   2
         0.5508481E+03   3   1   0
         0.1134382E+04   2   2   0
        -0.1160042E+04   1   3   0
         0.3557390E+03   3   0   1
         0.1017086E+02   2   1   1
         0.6969160E+03   1   2   1
         0.1258877E+04   0   3   1
         0.5197582E+02   2   0   2
         0.4328018E+03   1   1   2
        -0.1062101E+03   0   2   2
         0.5471942E+02   1   0   3
        -0.3211568E+02   0   1   3
        -0.1333515E+03   4   1   0
        -0.4993582E+03   3   2   0
        -0.3928734E+03   2   3   0
         0.1143333E+04   1   4   0
        -0.2252141E+03   4   0   1
         0.2641766E+03   3   1   1
         0.9964729E+02   2   2   1
        -0.3031654E+03   1   3   1
        -0.9919408E+03   0   4   1
```

(continua)

```
                0.7123589E+02   3   0   2
               -0.5026490E+03   2   1   2
               -0.2019268E+03   1   2   2
                0.1085769E+03   0   3   2
               -0.3689024E+02   2   0   3
                0.1259192E+03   1   1   3
                0.1398510E+02   0   2   3
               -0.5489551E+01   1   0   4
               -0.2721741E+02   0   1   4
               -0.8672702E+01   5   1   0
                0.9243725E+02   4   2   0
                0.9455495E+02   3   3   0
                0.4378558E+02   2   4   0
               -0.3012895E+03   1   5   0
                0.5875433E+02   5   0   1
               -0.7644942E+01   4   1   1
               -0.1544242E+03   3   2   1
                0.1174672E+01   2   3   1
                0.1881099E+02   1   4   1
                0.2699340E+03   0   5   1
               -0.3803897E+02   4   0   2
                0.2211474E+02   3   1   2
                0.1713199E+03   2   2   2
                0.6709845E+02   1   3   2
               -0.2999395E+02   0   4   2
                0.1285414E+02   3   0   3
                0.5457174E+02   2   1   3
               -0.9745477E+02   1   2   3
               -0.7197123E+01   0   3   3
               -0.1377454E+01   2   0   4
               -0.2120028E+02   1   1   4
                0.2071087E+02   0   2   4
                0.7003032E+00   1   0   5
                0.4229195E+01   0   1   5
(5)             4       0
(6)             0
(7)              0032        50000       50000       0.0050D0
(8)              10.0D0              10.0D0                  10.0D0
(9)              0          0.461D0
                 1          0.0D0
                 1          1.80D0
                 1          0.0D0
                 0          5.8510D0
                 1          0.0D0
                 0          0.119778D0
                 1          0.0D0
(10)               10.0D0
(11)               80629
```

In tabella 3 è mostrato un esempio della prima parte del file di output generato dall'esecuzione del programma traiettorie. In esso sono riportate le informazioni descriventi l'input letto dal programma quali le masse atomiche del sistema, il tipo di potenziale, il numero di tariettorie e la frequenza degli output intermedi, il passo di integrazione, la distanza di troncamento dell'integrazione e i criteri per selezionare le condizioni iniziali. Da ultimo la distanza atomo-diatomo iniziale e l'iniziatore della sequenza pseudorandom. Tali informazioni descrivono il calcolo richiesto e permettono il controllo dei valori specificati in input.

Tabella 3: Esempio di file di output del programma traiettorie (prima parte)

```
A+BC TRAJECTORIES, INPUT DATA

 MASSES- MA=    6.94100  MB=   18.99840  MC=    2.01410
 POTENTIAL PARAMETERS DAB,DBC,DAC
  137.59000  141.20000   58.00000
 BETAAB,BETABC,BETAAC,ROAB,ROBC,ROAC
 0.97076   2.19419   1.17086   1.56386   0.91681   1.59570
POTENTIAL TYPE= 4   HMF INDICATORS= 0 0   M+X2 INDICATOR= 0   MAPS= 0
TYPE OF STARTING CONDITIONS=   0 (0=SELECTED, 1=REA DQ+P)
 NO TRAJECTORIES 50000   MAX. CYCLES=50000   PRINT CYCLES=50000
TIME STEP SIZE=  0.01803 E-14 SECS
 CUTOFF R S (AB,BC,AC)=   10.000000   10.000000   10.000000
0PARAMETERS FOR SELECTION PROCEDURE
 VELOCITY
   0       0.4000000000D+01
 V ANGLE BETA
   1       0.0000000000D+00
 IMPACT PARAMETER
   1       0.2000000000D+01
 BC ANGLE THETA
   1       0.0000000000D+00
 BC VIBRATIONAL ENERGY
   0       0.4250000000D+01
 BC VIBRATIONAL PHASE
   1       0.0000000000D+00
 BC ROTATIONAL ENERGY
   0       0.0000000000D+00
 BC ROTATIONAL ORIENTATION
   1       0.0000000000D+00
 INITIAL SEPARATION= 10.000000
 RANDOM CHAIN INITIATOR=        80629
```

Nella seconda parte del file di output, mostrato in tabella 4, sono riportate le risultanze del calcolo sotto forma di analisi statistica delle traiettorie e di valore delle grandezze di scattering. In particolare, nell'esempio qui riportato, si dà il numero di traiettorie che hanno superato il numero massimo di cicli di integrazione prefissato,

il valore massimo del parametro d'urto ed infine per ogni canale (1=formazione di AB, 2=non reazione, 3=formazione di AC) il numero di traiettorie finite nel canale, la sezione d'urto, la distribuzione in θ, ϕ, μ, β dei reagenti, l'angolo di scattering dei prodotti e la relativa energia traslazionale, vibrazionale e rotazionale. Vengono altresì date informazioni dettagliate sulla popolazione dei singoli stati rotazionali e vibrazionali.

Tabella 4: Esempio di file di output del programma traiettorie (seconda parte)

TOTALS AND HISTOGRAMS-
TRAJECTORIES STOPPED= 7
MAXIMUM B= 2.00000

PRODUCT CHANNEL 1 NUMBER OF TRAJECTORIES= 232

TOTAL CROSS SECTION IN THIS CHANNEL = 0.058317 A**2

ENERGY RANGE FOR THIS HISTOGRAM= 5.00 KCAL

	B	COS(THETA)	PHI	FAZE	BETA	SCATT. ANGLE	TRANSLATION	VIBRATION	ROTATION
1	1	100	15	12	6	1	0	24	75
2	2	13	14	9	2	13	3	21	47
3	9	0	13	12	5	8	10	19	20
4	17	0	14	15	3	26	6	22	22
5	20	0	7	25	6	16	11	21	12
6	28	0	12	19	6	14	16	6	16
7	27	0	10	16	7	21	22	18	14
8	18	0	7	14	14	16	13	18	7
9	26	0	8	8	39	15	18	20	10
10	7	0	13	9	30	11	24	12	3
11	17	0	11	14	45	13	28	11	3
12	24	0	13	4	21	12	14	10	1
13	18	0	11	10	17	9	17	14	1
14	16	0	6	8	7	9	18	8	1
15	2	0	15	13	12	5	13	6	0
16	0	3	12	6	3	8	6	2	0
17	0	16	16	10	3	10	5	0	0
18	0	41	10	7	1	7	7	0	0
19	0	45	9	11	4	10	1	0	0
20	0	14	16	10	1	8	0	0	0
					OUT OF RANGE	0	0	0	
TOTALS=		184.97	-13.70	0.53	-79.15	709.64	18877.11	557.14	364.72	174.10
AVERAGES=		0.80	-0.06	0.00	-0.34	3.06	81.37	2.40	1.57	0.75

(continua)

QUANTUM ROTATIONAL DISTRIBUTION

0	0	50	0	100	0	150	0
1	1	51	0	101	0	151	0
2	3	52	0	102	0	152	0
3	12	53	0	103	0	153	0
4	15	54	0	104	0	154	0
5	16	55	0	105	0	155	0
6	7	56	0	106	0	156	0
7	21	57	0	107	0	157	0
8	17	58	0	108	0	158	0
9	15	59	0	109	0	159	0
10	10	60	0	110	0	160	0
11	7	61	0	111	0	161	0
13	11	63	0	113	0	163	0
14	13	64	0	114	0	164	0
15	7	65	0	115	0	165	0
16	5	66	0	116	0	166	0
17	7	67	0	117	0	167	0
18	10	68	0	118	0	168	0
19	10	69	0	119	0	169	0
20	9	70	0	120	0	170	0
21	3	71	0	121	0	171	0
22	6	72	0	122	0	172	0
23	6	73	0	123	0	173	0
24	4	74	0	124	0	174	0
25	4	75	0	125	0	175	0
26	1	76	0	126	0	176	0
27	1	77	0	127	0	177	0
28	1	78	0	128	0	178	0
29	1	79	0	129	0	179	0
30	0	80	0	130	0	180	0
31	0	81	0	131	0	181	0
32	0	82	0	132	0	182	0
33	0	83	0	133	0	183	0
34	0	84	0	134	0	184	0
35	0	85	0	135	0	185	0
36	0	86	0	136	0	186	0
37	0	87	0	137	0	187	0
38	0	88	0	138	0	188	0
39	0	89	0	139	0	189	0
40	0	90	0	140	0	190	0
41	0	91	0	141	0	191	0
42	0	92	0	142	0	192	0
43	0	93	0	143	0	193	0
44	0	94	0	144	0	194	0
45	0	95	0	145	0	195	0
46	0	96	0	146	0	196	0
47	0	97	0	147	0	197	0
48	0	98	0	148	0	198	0
49	0	99	0	149	0	199	0

OUT OF RANGE = 0

(continua)

```
QUANTUM VIBRATIONAL DISTRIBUTION
  0     184
  1      48

  3       0
  4       0
  5       0
  6       0
  7       0
  8       0
  9       0
 10       0
 11       0
 12       0
 13       0
 14       0
 15       0
 16       0
 17       0
 18       0
 19       0


OUT OF RANGE =  0
TOTALS AND HISTOGRAMS FOR TRAJECTORIES CROSSING THE REGION OF THE POTENTIAL WELL
TRAJECTORIES STOPPED=  6
TOTALS AND HISTOGRAMS FOR TRAJECTORIES CROSSING THE POTENTIAL WELL
TRAJECTORIES STOPPED=  1
END OF RANDOM NO STRING      921627458
```

Bibliografia

[1] *Non Equilibrium Processes in Partially Ionized Gases*, M. Capitelli and J.N. Barsdley ed. (Plenum, New York, 1990).

[2] G. C. Fox et al., Solving Problems on Concurrent Processors, Prentice Hall, 1988

[3] Geoffrey C. Fox, *Parallel computing comes of age: Supercomputer level parallel computations at Caltech,* Concurrency: practice and experience, Vol. 1(1), September 1989

[4] A. J. G. Hey *Experiment in MIMD Parallelism,* In Proc. of Int. Conf. PARLE 89, Eindhoven, The Netherlands, June 1989. LNCS 366 Springer-Verlag.

[5] R. Baraglia, R. Ferrini, D. Laforenza, R. Perego, O. Gervasi, A. Laganà, *Modelling and Evaluation of a Task Farm Chemical Application on MIMD Architectures,* High Performance Computing II, M. Durand and F. El Dabaghi

(Editors), Elsevier 1991

[6] R. Baraglia, "La programmazione degli elaboratori IBM multiprocessore shared memory", in Chimica Computazionale: fondamenti informatici, edito da A. Laganà, Dipartimento di Chimica, Università di Perugia, Perugia 1992, pp.111-127.

[7] R. Baraglia, D.Laforenza, R. Perego, A. Laganà, O. Gervasi, E. Garcia: "D + D2 quasiclassical constant calculations on parallel computers", *International Journal "Theoretica Chimica Acta"*, Springer-Verlag, November 1990, n.79 pag. 323-333.

[8] R. Baraglia, "La programmazione di architetture MIMD a memoria distribuita: gli ipercubi", in Chimica Computazionale: fondamenti informatici, edito da A. Laganà, Dipartimento di Chimica, Università di Perugia, Perugia 1992, pp.129-148.

[9] nCUBE Corporation, nCUBE 2 Programmer's Guide, 1990

[10] nCUBE Corporation, nCUBE 2 Programmer's Reference Manual, 1990

[11] R. Baraglia, R. Ferrini, D. Laforenza, R. Perego, *Performance evaluation of hypercube systems,* Parallel Computing: Problems, Methods and Applications, P.Messina and A. Murli (Editors), Elsevier 1992