Simple example of parallel code and how to run it on a cluster

Bruno Juliá-Díaz

U. Barcelona

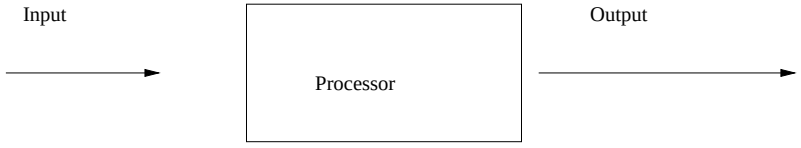
Motivation

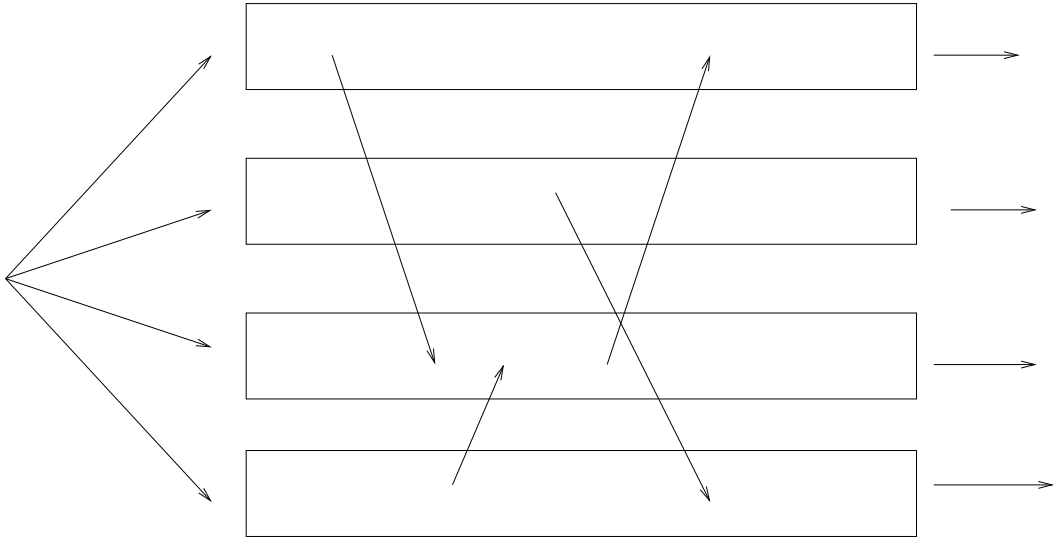
- Running in parallel
- one (very) simple example
- a second example with some physics

Running in parallel

Usually we run using one single processor

But we can also run in several of them at the same time!





Running in parallel (II)

- Tricky points:
  - The SAME code will run in ALL processors
  - How do we make that each of them computes different things?
  - in other words, how do we tell the code in which processor he is?
  - Maybe some information should be shared between processors
  - Example: They need to share preliminar results from time to time
- We need MPI (*message passing interface*)

Running in parallel (III)

- I ll use MPI with fortran (or c should be ok)
- This is quite standard

  - you can run in the Barcelona Supercomputer (BSC/CNS)
  - on a linux cluster
  - on your dual processor (using http://open-mpi.org)

Very Simple Example (I)

Very simple parallelization ⇒ Simple loop structures:

```
s.f          Mon Dec 18 15:32:08 2006          1

      do 2 i=1,2
      E=real(i)
      call computecrossection(E,CST)
      print*,CST
 2    continue
      end
      subroutine computecrossection(E,CST)
       CST=E**3
      end
```

We can easily parallelize it as each E value is independent from the other Es

Very Simple example (II)

```
sp.f        Mon Dec 18 15:27:22 2006        1
      include "mpif.h"
      call MPI_INIT(ierr)
      call MPI_COMM_RANK(MPI_COMM_WORLD, myid, ierr)
      call MPI_COMM_SIZE(MPI_COMM_WORLD, numprocs, ierr)

      do 2 i=1,2
      if (i.ne.myid+1) goto 2
      E=real(i)
      call computecrossection(E,CST)
      print*,"I am node", myid, "my value",CST
 2    continue
      CALL MPI_FINALIZE(IERR)
      end

      subroutine computecrossection(E,CST)
       CST=E**3
      end
```

Very Simple example (III)

The commands we have used are:

- include "mpif.h" Loads MPI library
- call MPI_COMM_INIT(ierr) Initialize MPI
- call MPI_COMM_RANK(MPI_COMM_WORLD, myid, ierr)
  gives the processor number, from 0 to Numprocs-1
- call MPI_COMM_SIZE(MPI_COMM_WORLD, numprocs, ierr)
  gives the number of proccessors
- call MPI_FINALIZE(ierr)
  Finalizes

Very Simple example (IV)

- key point We used a conditional to compute ONLY in a selected processor:
  if (i.ne.myid+1) goto 2

Four useful commands

Basic sending/receiving commands:

- CALL MPI_SEND(buff, count, MPI_TYPE, dest, tag, comm, ierr)
  - CALL MPI_SEND(ztsend,1,MPI_DOUBLE_COMPLEX,0,n1tag, MPI_COMM_WORLD,MPIERR)
- call MPI_RECV(rbuf, count, MPI_TYPE, source, tag, comm, status, ierr)
  - call MPI_RECV(ztrec,1,MPI_DOUBLE_COMPLEX,IRANK,n1tag, MPI_COMM_WORLD,MPISTAT,MPIERR)
- call MPI_BCAST(buff, count, MPI_TYPE, root, comm, ierr)
  - call MPI_BCAST(ct,1,MPI_DOUBLE_PRECISION,0,MPI_COMM_WORLD,ierr)
- CALL MPI_REDUCE(sendbuf, recbuf, count, MPI_TYPE, MPI_OP, root, comm, ierr)
  - call MPI_REDUCE(xmychi2,ct,1,MPI_DOUBLE_PRECISION,MPI_SUM,0, MPI_COMM_WORLD,ierr)

Executing the code

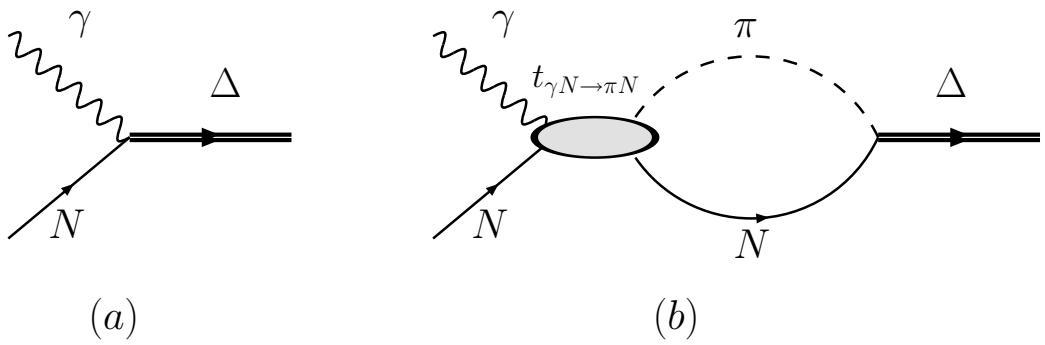
```
pa        Mon Dec 18 15:28:06 2006        1
bruno@melon:~/textos/barna_parallel06> more sp.f
      include "mpif.h"
      call MPI_INIT(ierr)
      call MPI_COMM_RANK(MPI_COMM_WORLD, myid, ierr)
      call MPI_COMM_SIZE(MPI_COMM_WORLD, numprocs, ierr)

      do 2 i=1,2
      if (i.ne.myid+1) goto 2
      E=real(i)
      call computecrossection(E,CST)
      print*,"I am node", myid, "my value",CST
 2    continue
      CALL MPI_FINALIZE(IERR)
      end

      subroutine computecrossection(E,CST)
       CST=E**3
      end

bruno@melon:~/textos/barna_parallel06> mpif77 sp.f
ifc: warning: The Intel Fortran driver is now named ifort.  You can suppress this message
 with '-quiet'
bruno@melon:~/textos/barna_parallel06> mpirun -np 2 a.out
 I am node           1 my value   8.000000
 I am node           0 my value   1.000000
bruno@melon:~/textos/barna_parallel06>
```

Self promotion

I am interested together with a group at Jefferson Lab
http://ebac-theory.jlab.org
in studying πN interactions in a full coupled channel formalism
The main aim is to elucidate the structure of nucleon resonances in photo and electroproduction processes

- Need to fit several parameters with some constraints
- Use χ^2 minimization
- Our function takes minutes to evaluate
- Thus: we decided to parallelize

Brief on the physics

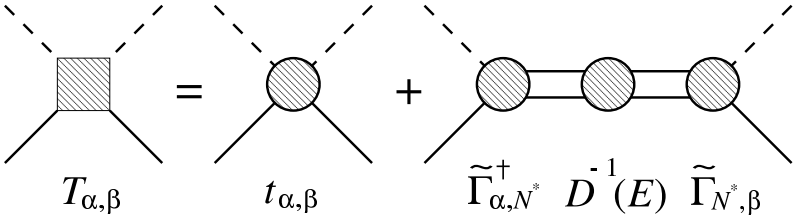


Relevant points:

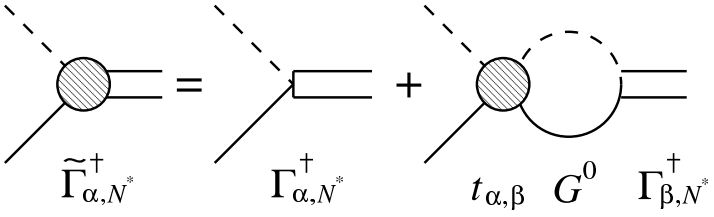
- Build a reaction theory to analyze the data
- Including most known phenomenology
- Coupled channel effects

Brief on the physics (II)

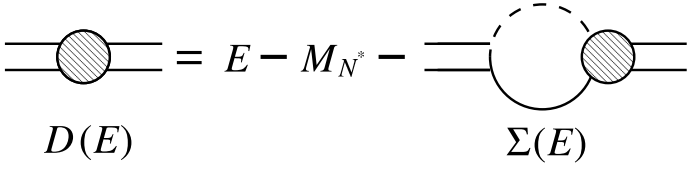
Non-resonant + resonant



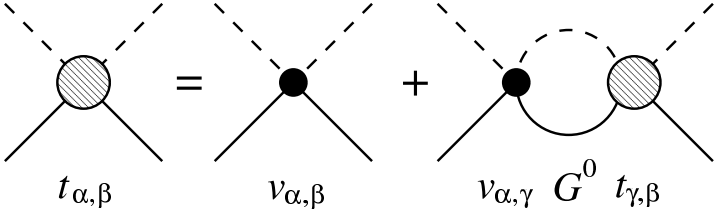
dressed resonant vertexes



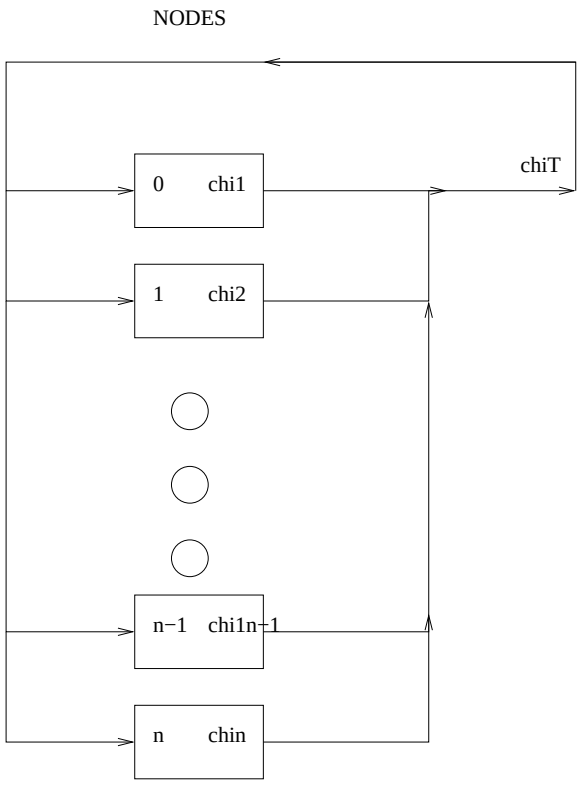
resonant self-energies



non-resonant tmatrix



Minimizating in parallel



- Our aim was to build a Parallel code
- by minimally changing an existing code

Coupled channel parallel code

The coupled channel code was parallelized: CCEBA and tested at the NERSC supercomputing facility (LBNL)

- At BSCwe have the project:
  "Dynamical Coupled Channel Analysis of Excited Baryons"
  with 100000 hours alloted.

Applying to BSC/CNS

The fifth fastest supercomputer is HERE
and sometimes it even runs...

- http://www.bsc.es
- application is EASY and on the WEB (next in April)
- Now each application is for 4 months
- Ask Assum or myself if you need help to fill the application

Some useful tutorials

some good tutorials and really useful guides can be found in http://www.nersc.gov

- http://www.nersc.gov/nusers/help/tutorials/mpi/intro/
- http://www.llnl.gov/computing/tutorials/parallel_comp

Summary & Outlook

- We have a nice supercomputer in Barcelona
- and a cluster in the department
- Lets use them! (properly)