Aplicaciones de PETSc-FEM en hidráulica

por Mario Storti, Rodrigo Paz, Lisandro Dalcín y Sergio Idelsohn

Centro Internacional de Métodos Numéricos

en Ingeniería - CIMEC

INTEC, (CONICET-UNL), Santa Fe, Argentina

<mstorti@intec.unl.edu.ar>

http://www.cimec.com.ar

26 de agosto de 2002

Necesidades de cálculo para un probl. hidráulico a gran escala

- **La diversidad de escalas de longitud y temporales lleva a problemas computacionales de gran magnitud.**
- **Tipicamente, tenemos escalas de longitud que van de los 100 de kms a densidades del orden de cientos de metros en el río (una relación 1:1000 o mayor).**
- **Utilizando la potencia del método de elementos finitos para refinar localmente se obtienen modelizaciones aceptables con mallas de entre 100,000 y 1,000,000 elementos triangulares.**
- **Considerando el problema multiacífero con varias capas por acuífero debe pensarse en un número d incgnitas de 10-20 por nodo.**
- **El uso de capas dentro de un acuífero permitirá recuperar el flujo vertical, lo cual servirá como dato para el problema del tranporte de contaminantes.**
- **De esta forma se llega fácilmente a un número de incgnitas de entre $10^6 - 10^7$.**

Nec. de cálculo p. un problema hidráulico a gran escala (cont...)

- **El programa de FEM consta básicamente de dos partes, evaluación del residuo y matrices, resolución del problema lineal asociado. Esto se hace para cada paso de tiempo.**
- **La evaluación del residuo y matrices lleva del orden de 1 sec/Kelem, para un solo acuífero en un procesador Pentium4. Estimando que el tiempo de evaluación sea lineal en función del número de acuíferos/subcapas estimamos un tiempo de 0.1 sec/Kelem/subcapa.**
- **Para un número de nodos de 10^6 y 10 capas esto da un tiempo de evaluación de 1000secs$\approx$ 17 m.**
- **Este tiempo debe ser multiplicado por el número de pasos de tiempo y, si el problema es no-lineal por el número de iteraciones del lazo de Newton-Raphson. El problema de flujo en medios porosos es no-lineal por el acuífero freático y por el acoplamiento con corrientes de superficie.**
- **Si consideramos un número de pasos de tiempo en el orden de 1000 legamos a tiempos de cálculo de varios días.**

Cálculo en CFD usando procesamiento distribuido en el CIMEC

- **Uso de clusters de tipo Beowulf (procesadores x86 corriendo GNU/Linux OS) (alta relación performance/precio)**
- **Esto implicaba la reescritura de nuestros programas de cálculo. (Uso de MPI+PETSc).**

Decidimos reescribir los programas con los siguientes lineamientos:

- **Desarrollar una librería general para escribir programas FEM incluyendo procesamiento en paralelo.**
- **Incluir técnicas de programación avanzadas (Programación Orientada a Objetos (OOP) en un entorno C/C++).**
- **Separar los niveles de programación (*"librería"* FEM de uso general/aplicaciones)**
- **Desde mediados de 1999 hemos orientado la estrategia de cálculo en CFD hacia el cálculo en paralelo usando clusters de tipo *"Beowulf"* (clusters de procesadores x86 corriendo GNU/Linux OS).**
- **Esto traía aparejado una reescritura de los programas para incluir llamadas a PETSc/MPI. Entonces decidimos reescribir nuestros programas con los siguientes lineamientos.**

Técnicas de computación de alta-performance (HPC)

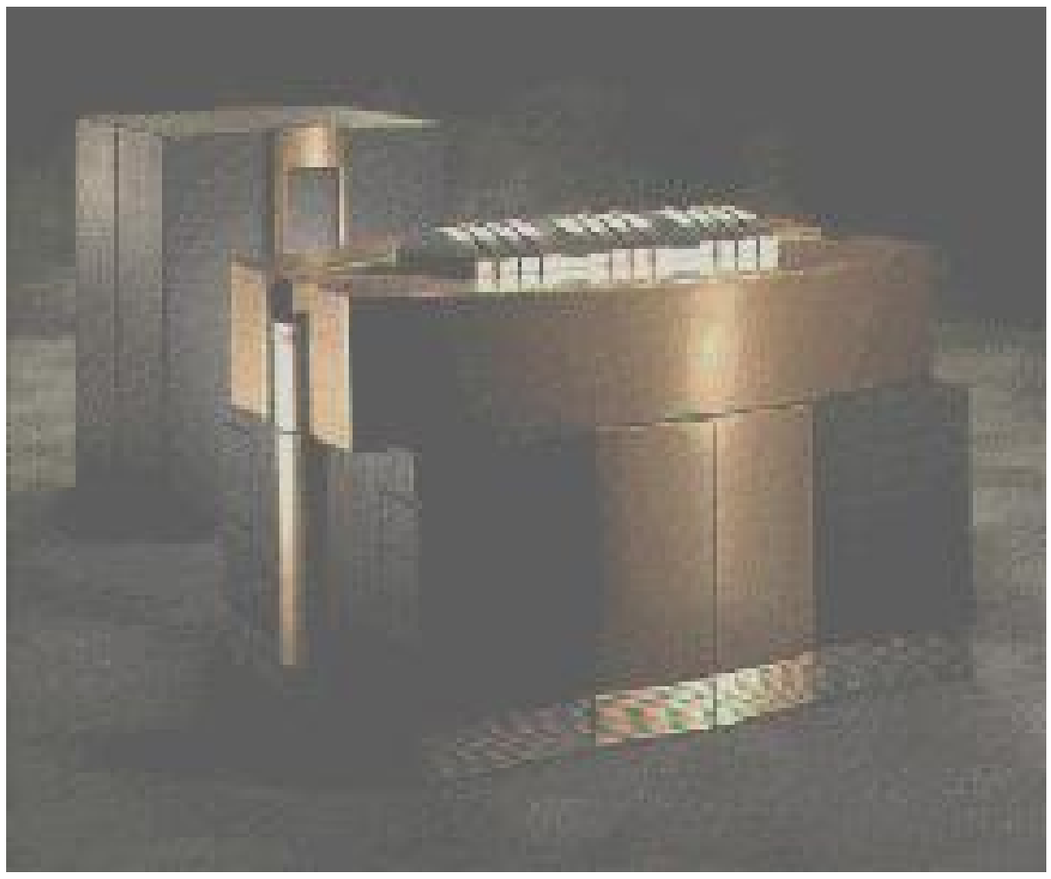
- **En la búsqueda de modelizaciones cada vez más precisas, se van desarrollando técnicas para incrementar la capacidad de cálculo.**
- **Una de las técnicas más comunes y prometedoras es la de dividir el problema en subproblemas más pequeños y resolver cada uno de los problemas en un procesador por separado. A esto se le da el nombre de *"procesamiento distribuido"*.**
- **En el mejor de los casos, si cada uno de los subproblemas es independiente del otro (están *"desacoplados"*) entonces el tiempo de cálculo en n procesadores es $T_n = T_1/n$.**
- **Debido a que, en general, los problemas no están desacoplados es necesario pasar cierta información de un procesador a otro. Es decir que $T_n = T_1/n + T_{\mathrm{comm}}$.**
- **El factor de ganancia al paralelizar se llama *"factor de aceleración"* (*"speedup"*) y se define como $S_n = T_1/T_n$. En el caso ideal en que el problema es completamente desacoplado entonces $S_n = n$. La *"eficiencia"* de la paralelización $\eta = S_n/n < 1$.**
- **Una lista de las más veloces computadoras (la lista de los *"Top 500"*) es mantenida por Jack Dongarra en `http://www.netlib.org`. Todas estas máquinas usan procesamiento distribuido, contando con hasta miles de procesadores.**

Evolución del hardware usado en el procesamiento distribuido

- **En el extremo “superior” estan las grandes firmas que venden supercomputadoras como Cray (hoy SGI) o IBM Asci-Red con miles de procesadores. Hoy las más comunes son las SGI Origin 2000.**
- **NOW/COW: A partir de los 80’s era común encontrar en los laboratorios de las universidades un cierto número de estaciones de trabajo (SUN/HP/DEC/SGI) y surgió la motivación de utilizar esta potencia de cálculo en corridas nocturnas. Surgieron las bibliotecas de *“Paso de Mensajes”* como PVM y MPI.**

Cray T94 en Centro de Supercomputación UFRGS, Porto Alegre Brasil

2 Procs. $\times$ 1.8 Gflops, Costo aprox. $ 5,000,000. `http://www.cesup.yfrgs.br`



Clementina II, SECTIP

- **Silicon Graphics Origin 2000, 40 procs.**
- **Costo aprox. $ 3,000,000**
- **`http://www.setcip.org.ar`**

Clusters Beowulf

- **Con el abaratamiento de las PC's y el advenimiento de software libre surgió la posibilidad de crear clusters de PC's completamente dedicados a cálculo. Estos son los llamados clusters Beowulf.**
- **De *"How to build a Beowulf"* (Sterling, T.L. et.al., MIT Press, 1999) un *"cluster Beowulf"* es *" Un cluster the 'mass-market commodity off-the-shelf' (*$\mathrm{M}^2\mathrm{COTS}$*) PC's interconectadas por tecnología LAN de bajo costo corriendo un OS open source de tipo Unix y ejecutando aplicaciones en paralelo con una librería de paso de mensajes estándar en la industria."***
  El *"Proyecto Beowulf"* fue desarrollado originalmente en el *Goddard Space Flight Center* (GSFC). También son populares los clusters con procesadores DEC/Alpha.

Top 500 cluster

- **Mantenido en `http://www.beowulf.org`**
- **Intitución: AIST - Computational Biology Research Center, Japan.**
  Nombre: CBRC Magi System. Integrador: NEC
  # de nodos: 520. # de procesadores: 1040.
  Performance: 967.20 Gflops. Tipo de red: Myrinet
- **Intitución: Real World Computing, Japan.**
  Nombre: CBRC RWC SCore Cluster III. Integrador: Self-made
  # de nodos: 512. # de procesadores: 1024.
  Performance: 955.40 Gflops. Tipo de red: Myrinet 2000
- **Intitución: Partnership American Museum of Natural History. USA**
  Nombre: ABACUS. Integrador: ???
  # de nodos: 281. # de procesadores: 564.
  Performance: 432.13 Gflops. Tipo de red: Fast Ethernet
- **Intitución: Chemnitz University of Technology. Germany**
  Nombre: CLiC. Integrador: Self-made
  # de nodos: 528. # de procesadores: 528.
  Performance: 422.40 Gflops. Tipo de red: Fast Ethernet

El cluster “Geronimo” en el CIMEC

- **El CIMEC (Centro Internacional de Métodos Numéricos en Ingeniería, ubicado en Santa Fe, dependiente del CONICET) desarrolla tareas de investigación en métodos numéricos desde 1982.**
- **Desde el año 1997 se vienen desarrollando experiencias en procesamiento distribuido. Originalmente en 4 procesadores DEC/Alpha 500/333 Mhz.**
- **Desde 1999 este esfuerzo se ha orientado a los cluster de PC corriendo bajo GNU/Linux. Actualmente, el cluster cuenta con 9 procesadores Pentium 4 1.4-1.7Ghz con 512 Mb RAM (Rambus) conectado a través de un switch Fast Ethernet (100 Mbit/sec, latency=$O(100)$) 3COM SuperStack 3300.**
- **La configuración del cluster es “disk-less” es decir que los nodos no tienen disco duro. Cargan el kernel de un diskette y montan el root filesystem via NFS en el server. El uso de la configuración disk-less permite una administración mucho más simple e implica un considerable ahorro ya que no requiere disco duro.**

Software utilizado

- **El software instalado en el cluster está basado en la distribución RedHat 7.1.**
- **Desarrollamos un código de elementos finitos escrito en C++ llamado PETSc-FEM. Actualmente tiene +35,000 líneas de código.**
- **PETSc-FEM también compila con diferentes versiones de GCC y EGCS hasta incluso la 2.96.**
- **Como librería de paso de mensajes usamos MPI (*"Message Passing Interface"*, `http://www.mcs.anl.gov/mpi`) en su implementacion MPICH (versión 1.0.2, `http://www.mcs.anl.gov/mpich`) desarrollados en el *Argonne National Laboratory* (ANL).**

Software utilizado. (cont.)

- **En general MPI no es llamado directamente sino a través de PETSc (versión 2.0.4) *Parallel Extensible Toolkit for Scientific Computations* que es un paquete orientado a métodos numéricos en procesamiento distribuido, y permite operaciones abstractas como definir vectores y matrices distribuidos y resolver los sistemas lineales asociados.**
- **Particionamiento de malla utilizando Metis. Este particionador de malla permite dividir el *"grafo dual"* (es decir áquel donde los elementos son vértices del grafo y los nodos son aristas que conectan los vértices) de conectividades en subdominios tratando de mantener la masa total de los vértices (esfuerzo computacional en computar los elementos) igual entre los diferentes subdominios manteniendo mínima la comunicación entre procesadores (la cual se define asignando un peso a las aristas del grafo dual). En el caso de realizar *"balance de carga"* la masa total de los vértices en cada procesador debe ser proporcional a la velocidad del procesador.**
  (`http://www.cs.umn.edu/~metis`, `http://www.cs.umn.edu/~karypis/metis`).

Software utilizado. (cont.)

- **Para el álgebra lineal se utiliza los paquetes estándar Lapack y Blas distribuidos por Netlib (`http://www.netlib.org/lapack/`).**
- **Contenedores abstractos de *Libretto* (`http://pobox.com/~aaronc/tech/libretto/`), Glib (http://www.gtk.org) y la C++ STL Template Library que viene con el compilador GNU gcc.**
- **Librería de matrices Newmat (`http://webnz.com/robert/`). Esta está siendo reemplazada por una liberría desarrollada por nosotros mismos llamada FastMat2.**
- **Cantidad de GNU – Open Source software como: TEX/LATEX, Emacs, Xfig, Tgif, Octave, Perl y muchos otros.**

Características de PETSc-FEM

- **GPL, accesible en `http://minerva.ceride.gov.ar/petscfem`, versión actual es petscfem-beta-3.01.**
- **Programa de elementos finitos de uso general y multi-física.**
- **Procesamiento en paralelo via uso de PETSc/MPI/Metis.**
- **Resolución de sistemas lineales via el Método de Descomposición de Dominios (DDM).**
- **Escrito en C++ con una concepción OOP, con especial énfasis en la eficiencia.**
- **Muchos tipos de elementos pueden ser usados en la misma aplicación, agrupando los elementos del mismo tipo en "elemsets".**
- **Propiedades de elementos pasadas a las rutinas de elementos via correspondencias *string*→*string* genéricas.**

Características de PETSc-FEM

- **Condiciones de contorno Dirichlet/Newman/mixtas/periódicas o restricciones generales.**
- **Cálculo de jacobianos numéricos.**
- **Actualmente implementados módulos de Navier-Stokes, hidrología sub-superficial, shallow-water, Euler, sistemas advectivos generalizados y ec. de Laplace.**
- **Perfiles de sistemas de ecuaciones calculados automáticamente.**
- **Librerías de matrices rápidas usando “caches” para las direcciones de memoria.**
- **Balance de carga**

Rendimiento en clusters heterogéneos

- **Si un procesador es más rápido y se asigna la misma cantidad de tarea a todos los procesadores independientemente de su velocidad de procesamiento, cada vez que se envía una tarea a todos los procesadores, los más rapidos deben esperar a que el más lento termine, de manera que la velocidad de procesamiento es a lo sumo n veces la del procesador más lento. Esto produce un deterioro en la performance del sistema para clusters heterogéneos. El concepto de speedup mencionado anteriormente debe ser extendido a grupos heterogéneos de procesadores.**

Rendimiento en clusters heterogéneos (cont.)

- **El trabajo W (un cierto número de operaciones) es dividido en n partes iguales $W_i = W/n$**
- **Cada procesador tarda un tiempo $t_i = W_i/s_i = W/ns_i$ donde s_i es la velocidad del procesador i (por ejemplo en Mflops). El hecho de que los t_i sean distintos entre si indica una pérdida de eficiencia.**
- **El tiempo total transcurrido es el mayor de los t_i, que corresponde al menor s_i:**

$$T_n = \max_i t_i = \frac{W}{n \min_i s_i} \quad (1)$$

- **El tiempo T_i podemos tomarlo como el obtenido en el más rápido, es decir**

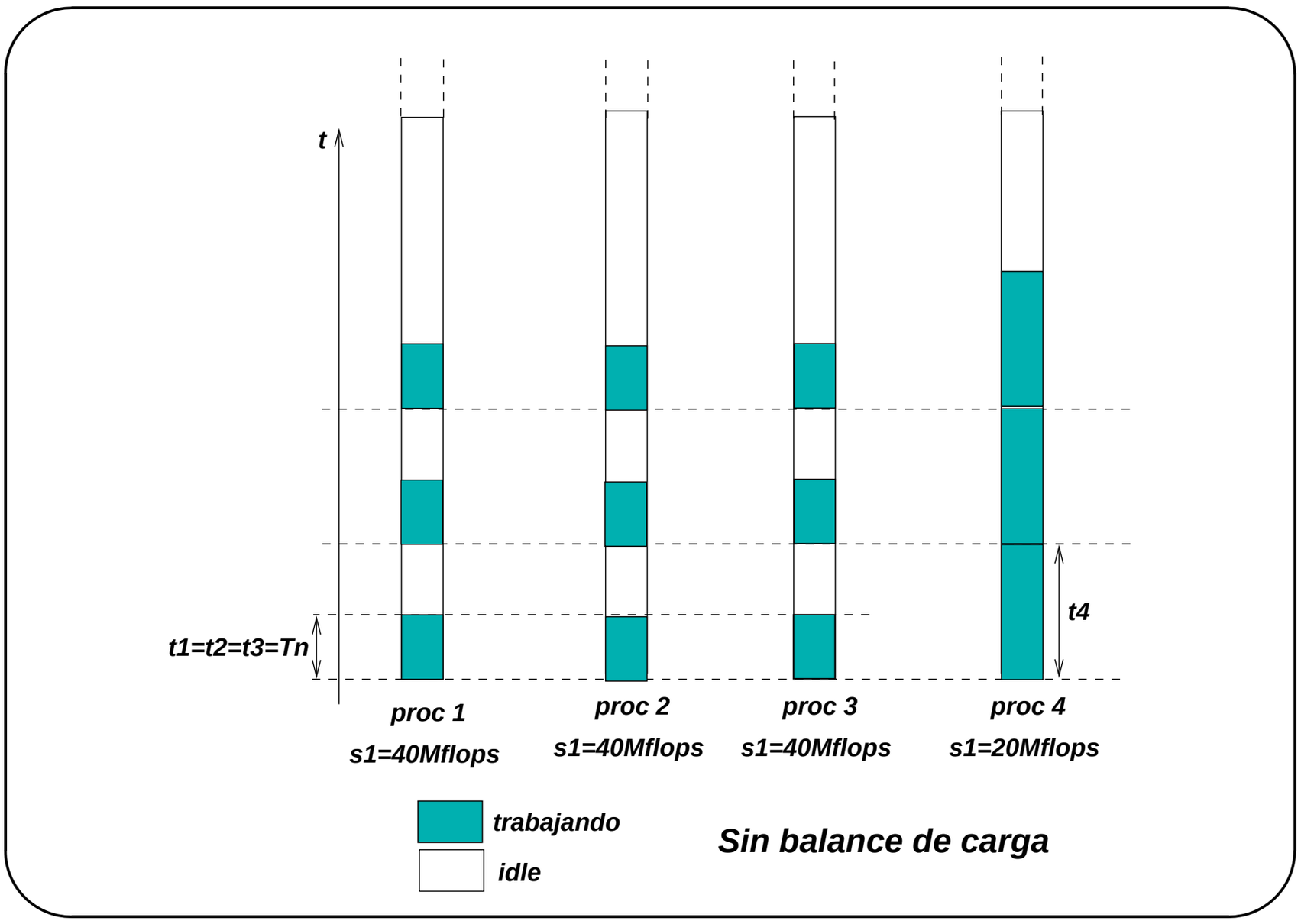
$$T_1 = \min t_i = \frac{W}{\max_i s_i} \quad (2)$$

Rendimiento en clusters heterogéneos (cont.)

- **El speedup resulta ser entonces**

$$S_n = \frac{T_1}{T_n} = \frac{W}{\max_i s_i} \Big/ \frac{W}{n \min_i s_i} = n \frac{\min_i s_i}{\max_i s_i} \tag{3}$$

El *"factor de desbalance"* $\min_i s_i / \max_i s_i$ **puede llegar en nuestro a** 0.7 **con lo cual el speedup teórico puede llegar a bajar de** 11 **a** 7.7**, que casi es igual a tomar solamente los 7 procesadores más rápidos sin incluir los más lentos. Esto sin tener en cuenta el deterioro en el speedup por los tiempos de comunicación.**


t
t1=t2=t3=Tn
t4
proc 1
s1=40Mflops
proc 2
s1=40Mflops
proc 3
s1=40Mflops
proc 4
s1=20Mflops
trabajando
idle
Sin balance de carga

Balance de carga

- **Si distribuimos la carga en forma proporcional a la velocidad de procesamiento**

$$W_i = W \frac{s_i}{\sum_j s_j}, \quad \sum_j W_j = W \tag{4}$$

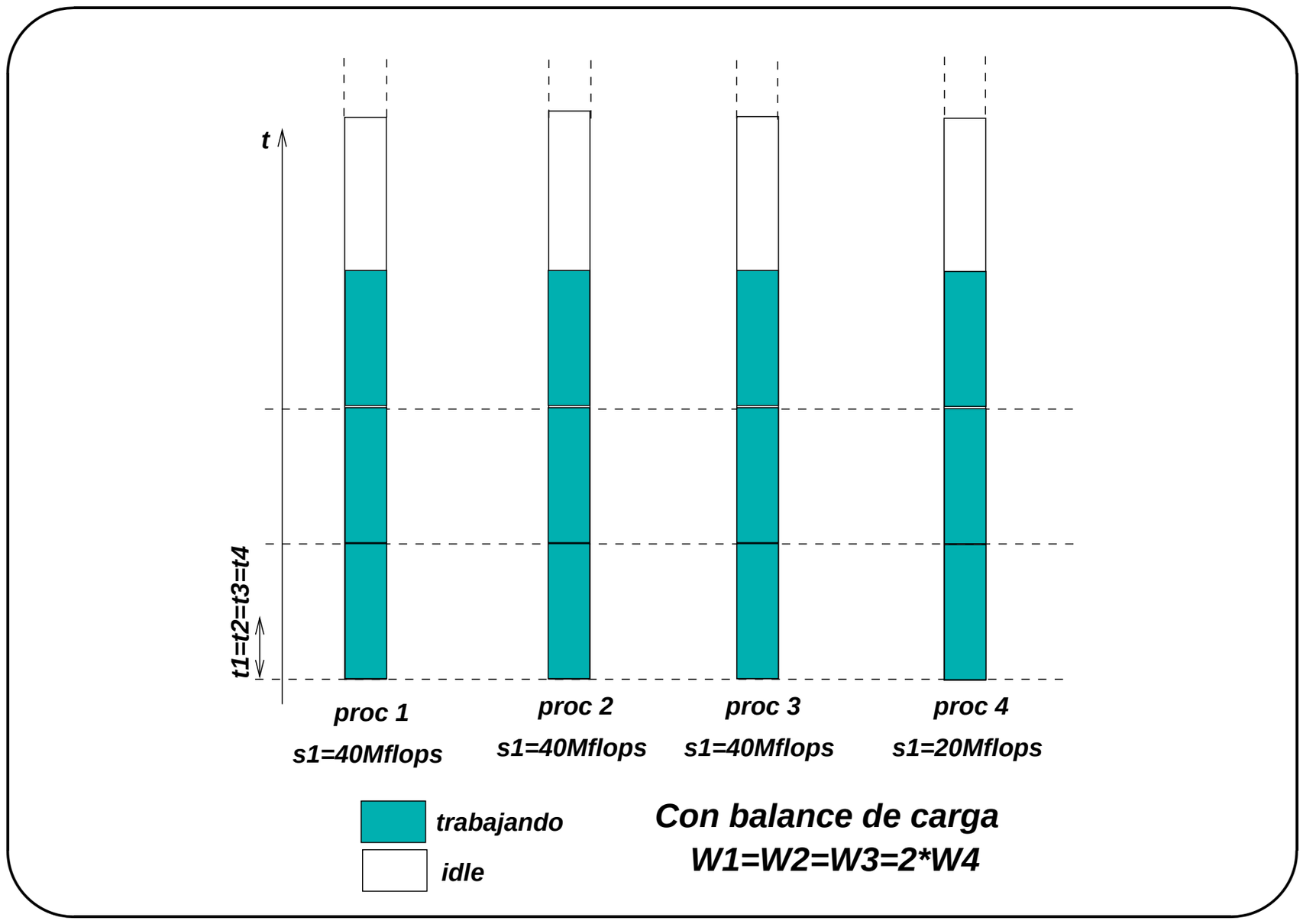
- **El tiempo necesario en cada procesador es**

$$t_i = \frac{W_i}{s_i} = \frac{W}{\sum_j s_j} \quad \textbf{(independiente de } i\textbf{!!!)} \tag{5}$$

- **El speedup ahora es**

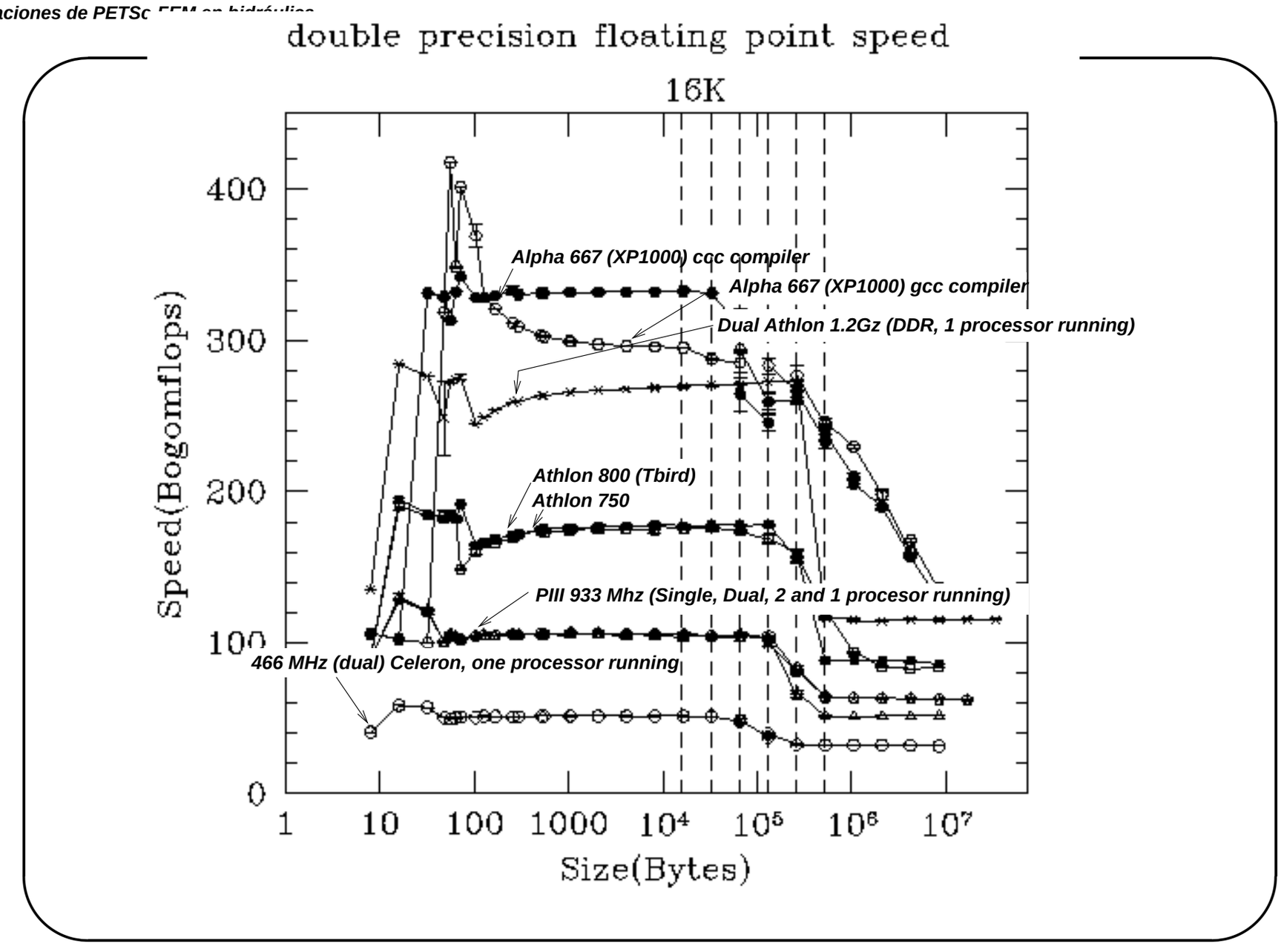
$$S_n = \frac{T_1}{T_n} = (W / \max_j s_j) / \left(\frac{W}{\sum_j s_j} \right) = \frac{\sum_j s_j}{\max_j s_j} \tag{6}$$

Este es el máximo speedup teórico alcanzable en clusters heterogéneos.


t
t1=t2=t3=t4
proc 1
s1=40Mflops
proc 2
s1=40Mflops
proc 3
s1=40Mflops
proc 4
s1=20Mflops
trabajando
idle
Con balance de carga
W1=W2=W3=2*W4

Balance de carga estático en PETSc-FEM

- **Actualmente PETSc-FEM permite balancear en forma estática la carga, esto es, una vez determinada la velocidad de los procesadores estos son leidos antes de comenzar la ejecución y la partición de la malla se realiza de manera de mantener en cada procesador la misma cantidad de “carga” (número de elementos).**
- **Desventajas del balance estático: Como tomar la velocidad del procesador?**
- **El procesador parece tener diferentes velocidades dependiendo del compromiso entre número de operaciones realizadas y cantidad de información que tiene que fluir desde la memoria al procesador.**
- **A veces el trabajo a realizar en el procesador no es el mismo para todos los elementos.**


double precision floating point speed
16K
Speed(Bogomflops)
400
300
200
100
0
1
10
100
1000
10⁴
10⁵
10⁶
10⁷
Size(Bytes)
Alpha 667 (XP1000) ccc compiler
Alpha 667 (XP1000) gcc compiler
Dual Athlon 1.2Gz (DDR, 1 processor running)
Athlon 800 (Tbird)
Athlon 750
PIII 933 Mhz (Single, Dual, 2 and 1 procesor running)
466 MHz (dual) Celeron, one processor running

PETSc-FEM / Hydrology module

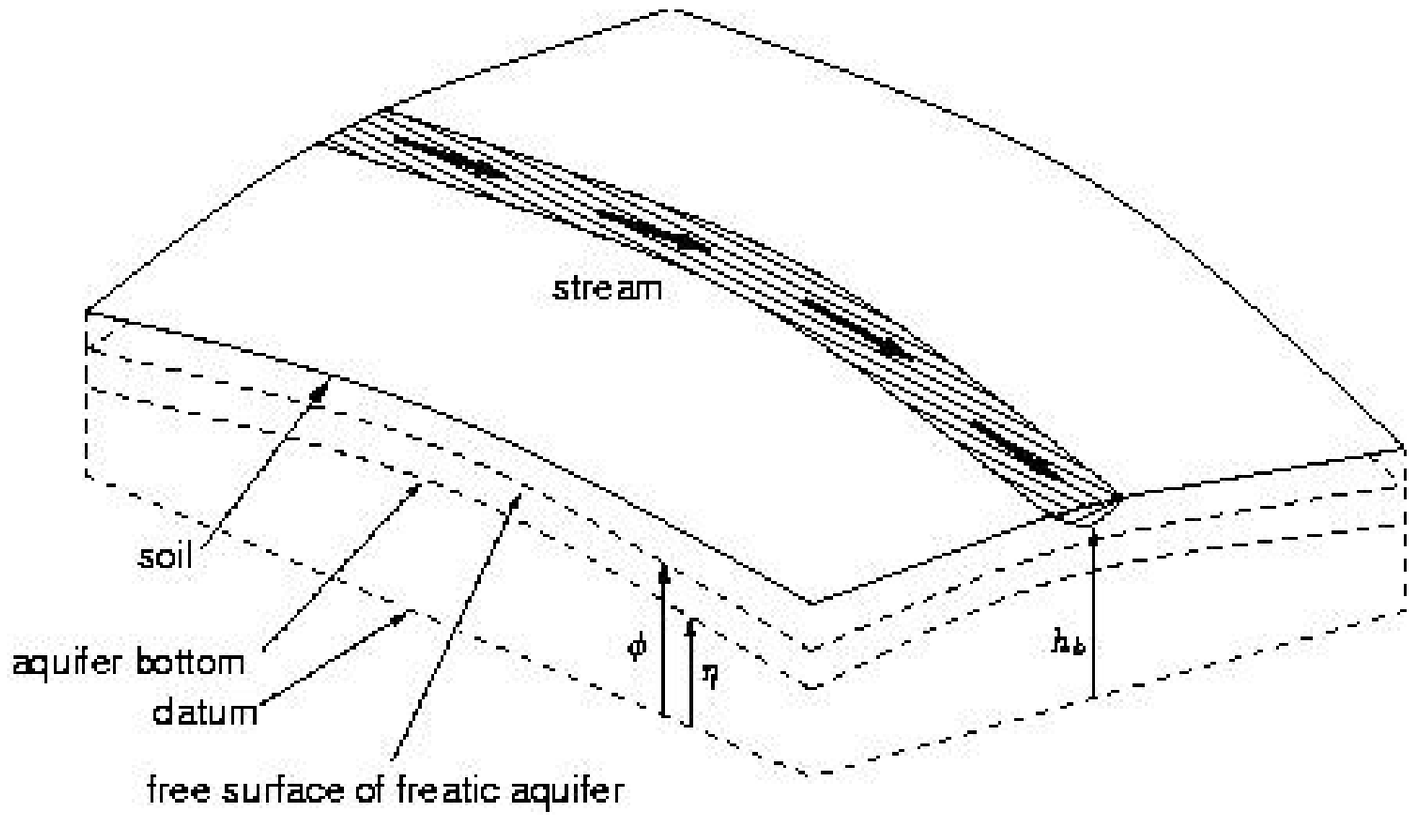


Figura 1: Aquifer/stream system.

PETSc-FEM / Hydrology module (cont...)

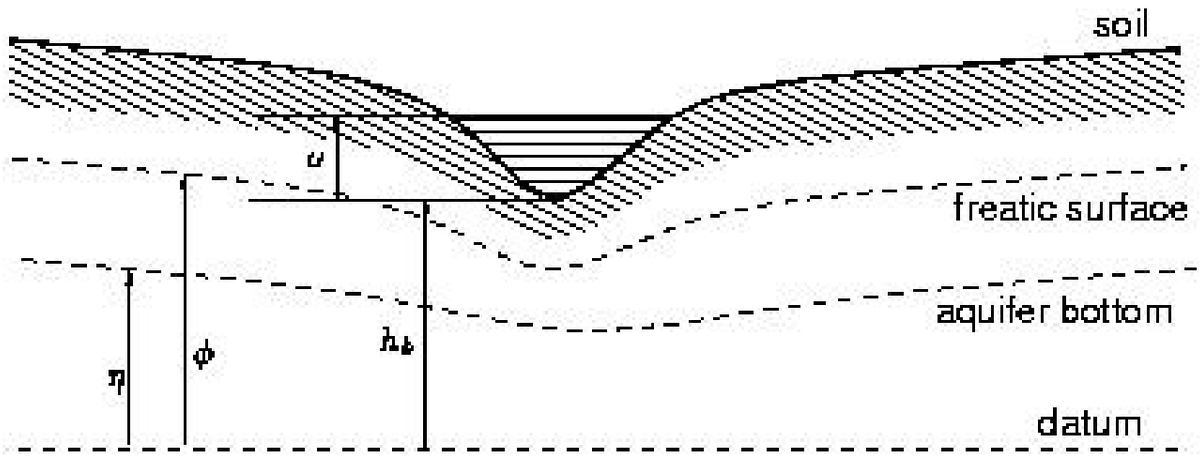


Figura 2: Aquifer/stream system. Transverse 2D view

PETSc-FEM / Hydrology module (cont...)

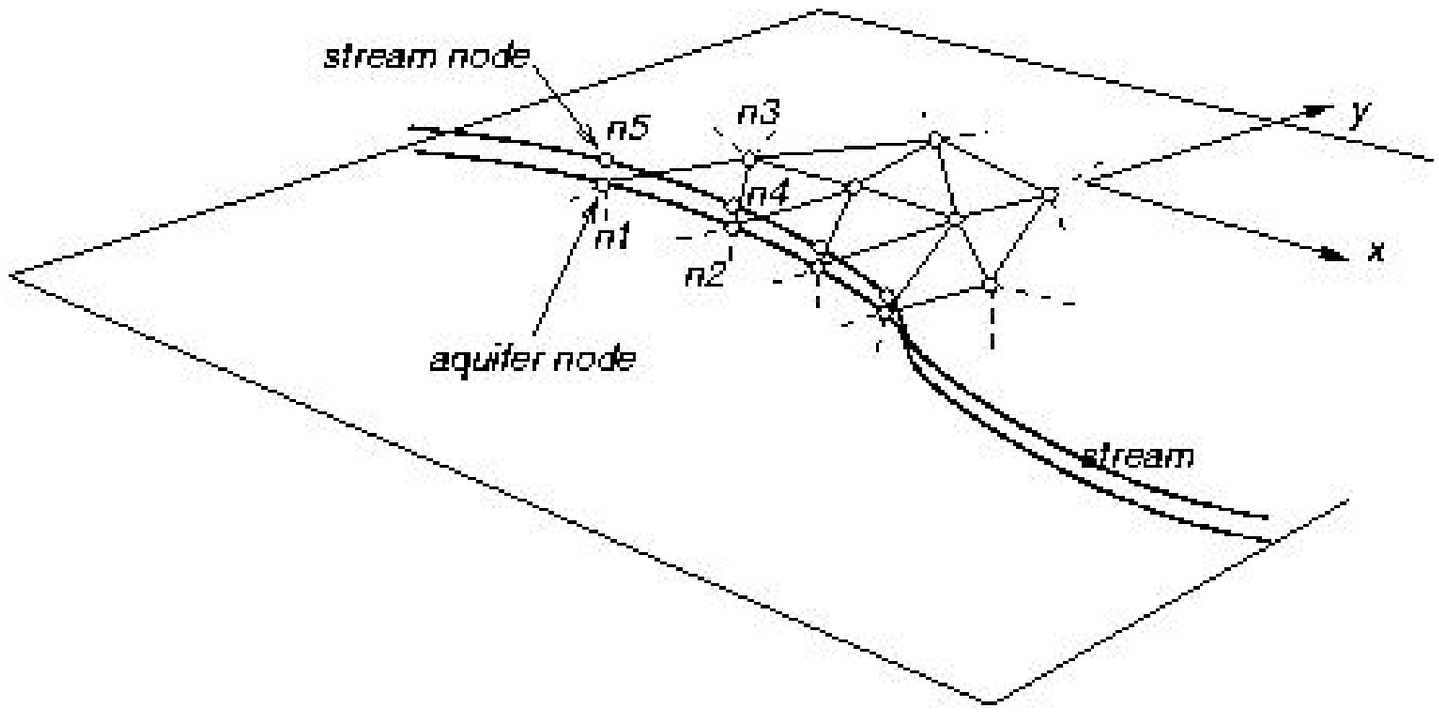


Figura 3: Aquifer/stream system. Discretization.

PETSc-FEM / Hydrology module (cont...)

This module solves the problem of subsurface flow in a free aquifer, coupled with a surface net of 1D streams. To model such system three elemsets must be used: an `aquifer` system representing the subsurface aquifer, a `stream` elemset representing the 1D stream and a `stream_loss` elemset representing the losses from the stream to the aquifer (or vice versa) see figures 1 and 2.

PETSc-FEM / Hydrology module (cont...)

The `aquifer` elemset is a 2D elemset with triangle or quadrangle elements (see figure 3). A per-element property `eta` represents the height of the aquifer bottom to a given datum. The corresponding unknown for each node is the piezometric height or the level of the freatic surface at that point ϕ. On the other hand, the `stream` elemset represents a 1D stream of water. It has its own nodes, separate from the aquifer nodes, whose coordinates must coincide with some corresponding node in the aquifer. For instance, the aquifer element in the figure is connected to nodes $n1$, $n2$ and $n3$, while the stream element is connected to nodes $n4$ and $n5$. $n1$ and $n4$ have the same coordinates (but different unknowns) and also $n2$ and $n5$. A node constant field (so called "H-fields") represents the stream bottom height h_b, with reference to the datum. So that, normally, we have for each node two coordinates and the stream bottom height. (`ndim=2 nu=3 ndof=1`). The unknown for these nodes is the height u of the stream free water surface with reference to the stream bottom. The channel shape and friction model and coefficients are entered via properties described below. If the stream level is above the freatic aquifer level ($h_b + u > \phi$) then the stream losses water to the aquifer and vice versa.

PETSc-FEM / Hydrology module (cont...)

The equation for the aquifer integrated in th verical direction is

$$\frac{\partial}{\partial t}\left(S(\phi-\eta)\phi\right) = \nabla\cdot\left(K(\phi-\eta)\nabla\phi\right) + \sum G_a \tag{7}$$

where S is the storativity and G is a source term, due to rain, losses from streams or other aquifers.

The equation for the stream is, according to the *"Kinematic Wave Model" KWM* approach ([?]),

$$\frac{\partial A(u)}{\partial t} + \frac{\partial Q(A(u))}{\partial s} = G_s \tag{8}$$

PETSc-FEM / Hydrology module (cont...)

Where u is the unknown field that represents the height of the water in the channel with respect to the channel bottom as a function of time and a linear arc coordinate along the stream, A is the transverse cross section of the stream and depends, through the geometry of the channel, on the channel water height u. Q is the flow rate and, under the KWM model is a function only of A through the friction law.

$$Q = \gamma A^m \tag{9}$$

where $\gamma = C_h\, S^{1/2}\, P^{-1}$ and $m = 3/2$ for the Chèzy friction model, and $\gamma = \bar{a}\, n^{-1}\, S^{1/2}\, P^{-2/3}$ and $m = 5/3$ for the Manning model, where $S = (\mathrm{dh_b}/\mathrm{ds})$ is the slope of the stream bottom, P is the wetted perimeter, and C_h, $\bar{a}$ and n are model constants. G_s represent the gain or loss of the stream, and the main component is the loss to the aquifer

$$G_s = P/R_f(\phi - h_b - u) \tag{10}$$

where R_f is the resistivity factor per unit arc length of the perimeter. The corresponding gain to the aquifer is

$$G_a = -G_s\, \delta_{\Gamma_s} \tag{11}$$

where Γ_s represents the planar curve of the stream and δ_{Γ_s} is a Dirac's delta distribution with a unit intensity per unit length, i.e.

$$\int f(\mathbf{x})\, \delta_{\Gamma_s} = \int_0^L f(\mathbf{x}(s))\, \mathrm{d}s \tag{12}$$

PETSc-FEM / Hydrology module (cont...)

The `stream_loss` elemset represents this loss, and a typical discretization is shown in figure 3. The stream loss element is connected to two nodes on the stream and two on the aquifer and must be entered in that order in the element connectivity table, for instance

```
elemset stream_loss 4
<.... elemset properties...>
__END__HASH__
...
<n5> <n4> <n1> <n2>
...
__END__ELEMSET__
```

Related Options

- `double width` (default=`<required>`): **Width of the channel**

PETSc-FEM / Hydrology module (cont...)

- `double radius` (default=`<required>`):

  Radius of the channel

- `double Ch` (default=`<required>`):

  Chezy roughness coefficient

- `double Rf` (default= `1.`): **Resistivity (including perimeter) of the stream to loss to the aquifer.**

- `int impermeable` (default=`0`):

  Flag whether the element is impermeable ($R_f \to \infty$) or not.

- `string shape` (default=`string("undefined")`):

  Choose channel section shape, may be `circular` **or** `rectangular`

PETSc-FEM / Hydrology module (cont...)

- `string friction_law` (default=`string("undefined")`):

  Choose friction law, may be `manning` **or** `chezy`

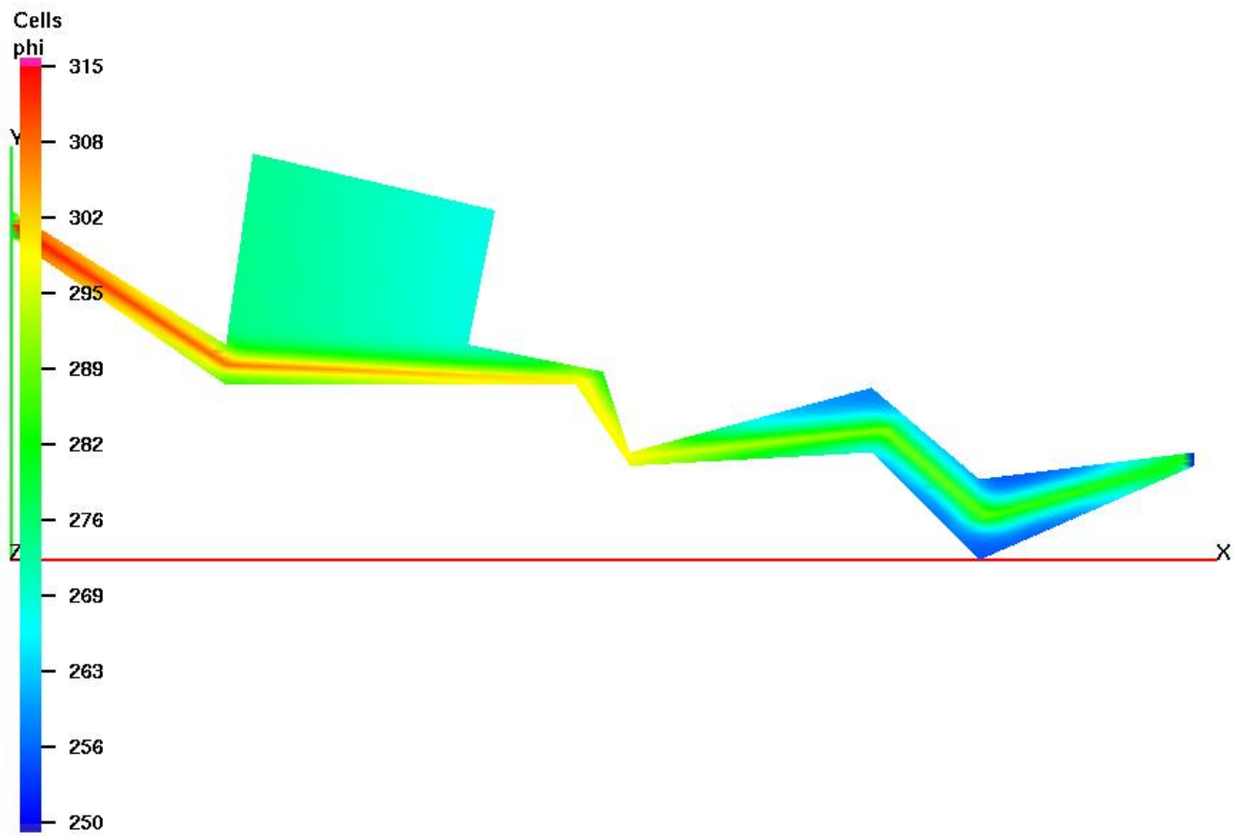
- `double roughness` (default=`1.`):

  Roughness coefficient for the Manning formula (a.k.a. n**)**

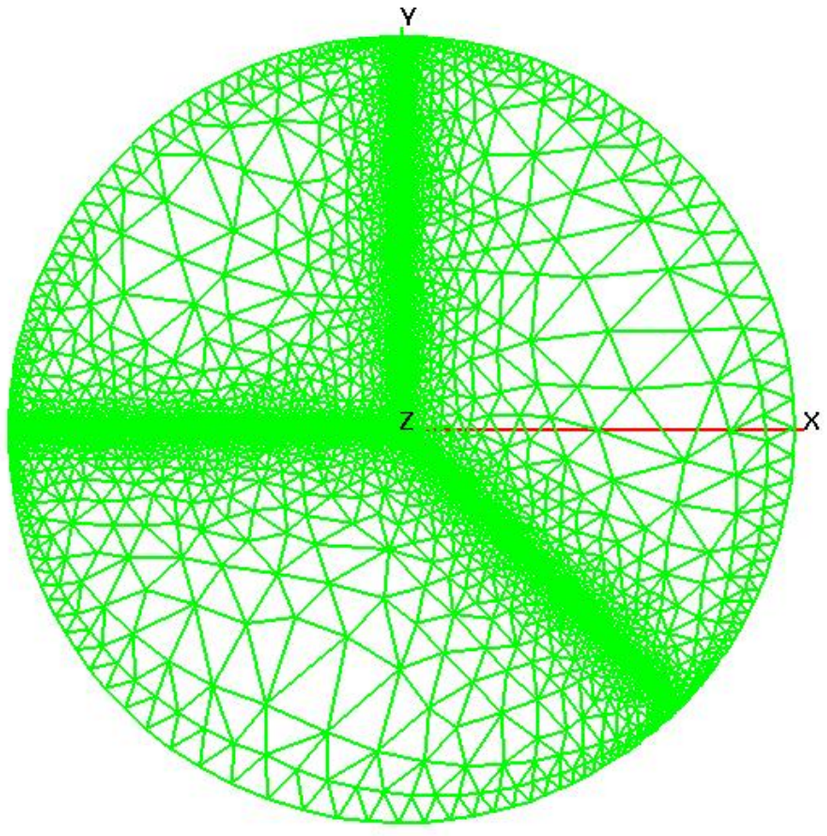
- `double a_bar` (default=`1.`):

  Unit conversion factor for Manning friction law.

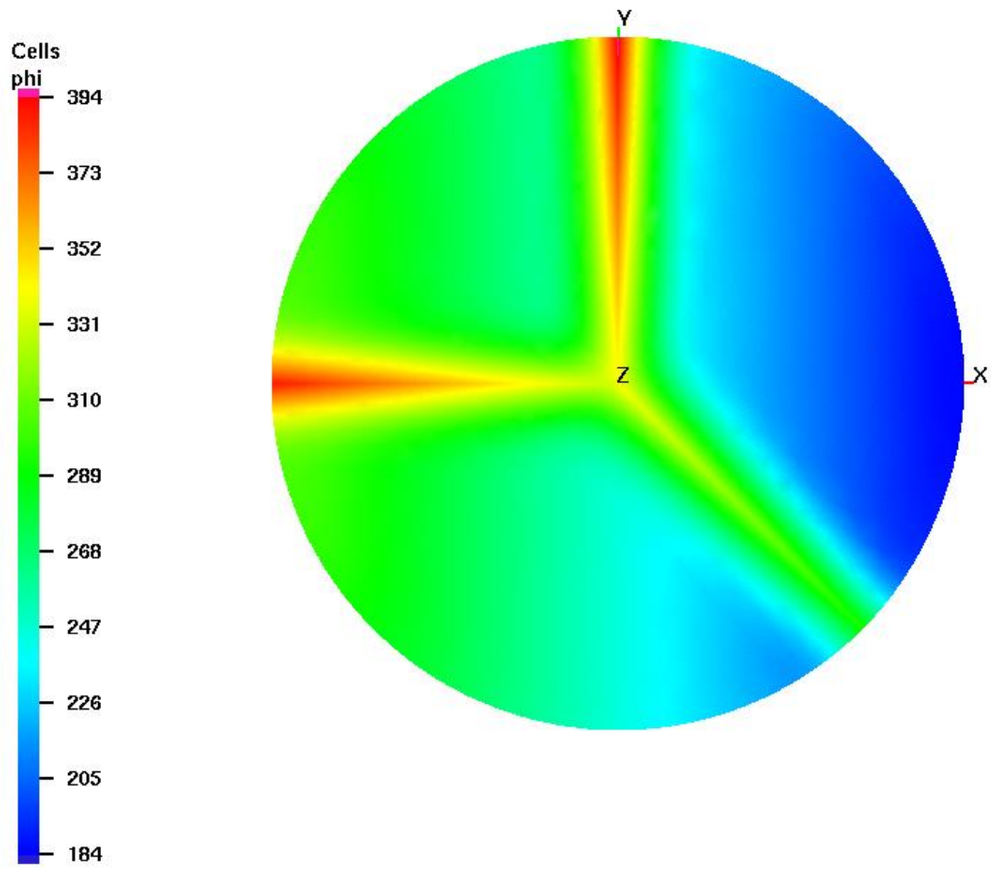
PETSc-FEM / Hydrology module (cont...)



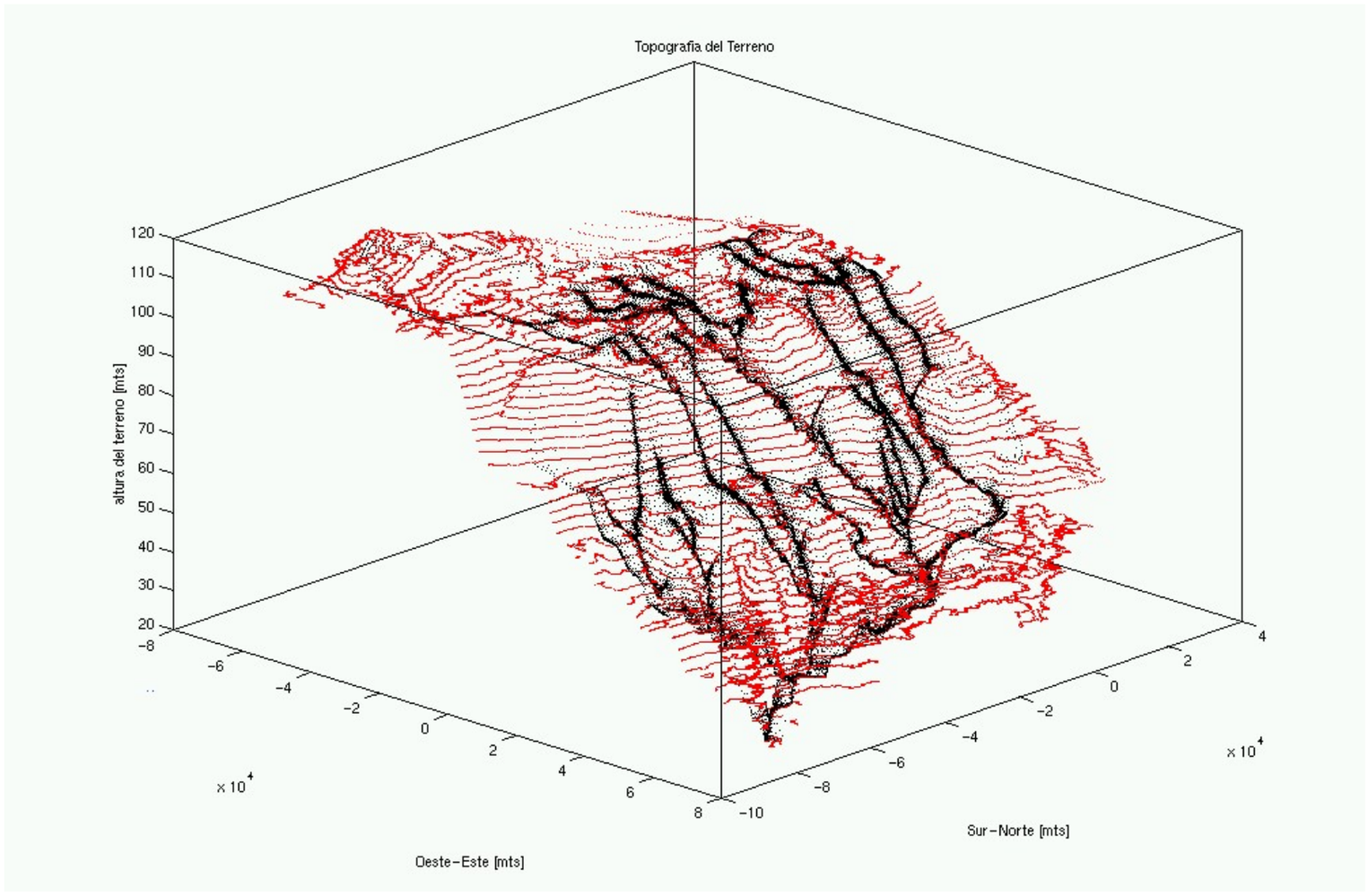
Caso test. Malla de FEM



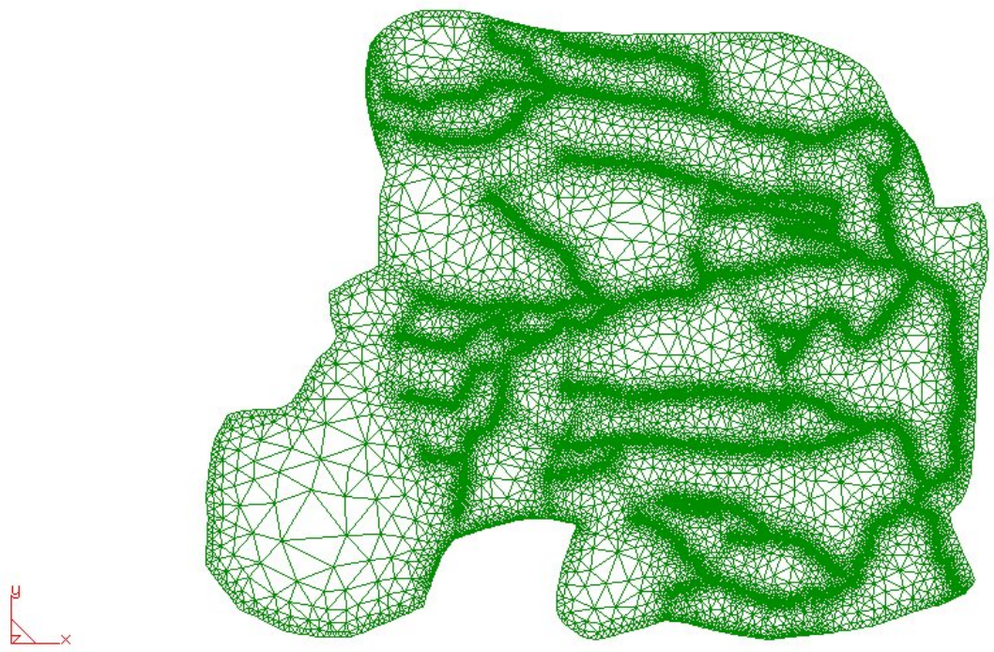
Caso test. Resultados



Cuenca del Cululu. Topografía y red de drenaje.



Cuenca del Cululu. Malla de elementos finitos.



Trabajo futuro

- **Incorporar varios acuíferos y subcapas**
- **Transporte de contaminantes acoplado con el flujo**
- **Acoplar el modelo de shallow water con el flujo subterráneo**