

AMIGA

1. Scenario Overview

Give an introduction to the scenario, making the reader familiar with goals, main purpose and functional overview.

1.1 Background and Purpose

AMIGA is a code for cosmological N-body simulations, i.e. simulating the whole Universe in a computer. It is purely grid-based and uses an arbitrary number of particles (each having its individual mass). The interparticle forces due to gravitational interactions are computed by solving Poisson's equation on a hierarchy of grids: while the whole computational domain (i.e. a cubic box representing the infinite Universe via periodic boundary conditions) is covered by a regular mesh, in high density regions the code places finer grids in a recursive manner that freely adapts to the actual particle distribution. These "refinement grids" can have arbitrary shape and entail an adaptive force softening during the course of the simulation.

1.2 More information

<http://www.aip.de/People/AKnebe/MLAPM>
bzw.
<http://www.aip.de/People/AKnebe/AMIGA>

2. Current Scenario description

Describe the current scenario including as much details as possible.

Note, there is no special section to describe workflows/pipelines or details about a phased execution of a program. If your scenario is a workflow/pipeline OR your application is executed in several phases, describe EACH step/part covering the sections 2.1 - 2.8. In addition you must describe how these steps/parts are interrelated to each other (in Section 2.8).

2.1 Environment

2.1.1 Hardware

Describe the hardware resources that are currently used.

- Processing

- . so far, AMIGA runs only on a single CPU (not parallelized!)
- . for scientific simulations it though requires large amounts of RAM (ranging from 2GB to 32GB depending on the specific problem)

- Storage

- . AMIGA stores snapshots of the whole simulation (number of such output files specified by user) in a path also specified by the user, e.g. locally or via nfs

- Network

- . not specified as AMIGA is a serial code

- Describe special hardware or other hardware resources that are relevant for the scenario.

- . no special hardware required

2.1.2 Software

- Describe used software such as operating system, software libraries, e.g. HDF5-plugin for GridFTP, ...

- . no special software required
- . runs on every platform (Mac OS X, Windows, Linux/Unix) as long as there is some C compiler

- What programming language is used and what compiler/linker version is required?

- . ANSI C, e.g. should work with every C compiler

- How is the program deployed?

- . source code:
the download package contains not only the source code for running cosmological simulations but also a suite of analysis and other tools.

- How is the program compiled?

- . make
the source plus all (analysis) tools are more or less "hidden" from the user in sub-directories: the user only "sees" the master Makefile and can make all binaries from the top-level directory
- . however, depending on the scientific problem the user may wish to play with (technical) parameters and hence needs to adjust not only the Makefile but also a file param.h in AMIGA's src/ directory

- State the program license and any commercial 3rd party licenses.

- . GNU General Public License

2.2 User Interaction

Describe the user interaction necessary for starting the program and additional interaction with a running program.

2.2.1 Initiation

- Describe how the program is started and any steps needed before the actual initiation.

1. the user needs to obtain a file with the initial conditions: this is a highly non-trivial problem in itself! but for the description of the "AMIGA Use Case" I assume that this file already exists...
2. the user needs to adjust an input file for AMIGA which contains

the following information:

- . the name of the input file containing the initial conditions
- . a prefix for the output files (incl. the full path)
- . three technical parameter (number of domain grid cells, two refinement criteria)
- . the final time at which AMIGA should terminate (in terms of cosmological redshift)
- . how often to write a dump file (this is simply an output file with a fixed name and hence the old dump file will be overwritten when writing a new dump file. the dump file just acts as a backup copy of the simulations in case the code crashes...)
- . the required number of output files
- . the times when to write such an output file

- compilation (cf. Section 2.1.2),

- . unless the user likes to fiddle with technical parameters, a simple "make" creates the binary
- . however, one or two flags need to be set in the Makefile depending on the nature of the initial conditions

- Where is the program executed?

- . single PC

- How is the program initiated?

- . command line:
\$ AMIGA < AMIGA.input
where AMIGA.input contains above specified parameters...

2.2.2 Monitoring/Steering/Visualization during the run-time of the program

Definitions:

Monitoring - Information retrieval regarding the state of the program. For example, "Program is running" or more application specific information such as "Current simulation iteration is 42".

Steering - Remote alteration of the programs state. For example, program stop or application specific information like "at iteration 42, set parameter x = 78".

Visualization - Remote access of the application data needed for visualization of, for example, a simulation.

- What type of data is produced by the program during run-time used for monitoring/steering/visualization?

monitoring

- . upon startup AMIGA dumps some information to stdout and stderr (startup depends upon the simulation again, but should not take longer than a couple of minutes to an hour!)
- . during run-time AMIGA writes all communication and information into one single logfile:
 - names of input file, output files, dump file, etc.
 - parameters used throughout the run
 - current state of the simulation (i.e. time step, accuracy, etc.)

- problems encountered during the run

steering

- . there is only one interaction possible: the user can "shut-down" AMIGA: if a (empty) file called "terminateAMIGA" exists (interactively created by the user), AMIGA finishes the current step, writes an output file, and exists

visualisation

- . post-processing of output files via provided tools and supplied IDL routines

- What methods/tools exists for accessing data produced by the program during run-time?

- . only post-processing via analysis software

- Does your application support any standard for monitoring/steering?

- . no

- Describe any security measures related to program access for monitoring/steering/visualization.

- . none

- Who can access the running program OR run-time produced monitoring data?

- . owner and system administrators

- From where can run-time produced monitoring data be accessed?

- . local access via shell

- How is the program termination detected?

- . normal stop of program (exit).
- . errors are handled by the program and written to logfile

- How much monitoring data and how often is monitoring data transferred during a program run (min/max/avg)?

- . transfer of a couple of bytes during run-time to logfile after every step

- Does your program generate metadata and stores this externally (e.g. in a catalog)?

- . all relevant run-time information is written to one logfile

- Who accesses this metadata? From where? Does your program access metadata generated by other programs?

- . does not apply

- How many executions/jobs must be monitored/steered in parallel? By how many users?

. AMIGA is a serial code and hence every executable creates its own logfile

2.3 Input

2.3.1 Parameters

Describe the program parameters in detail.

1. the user needs to adjust an input file for AMIGA which contains the following information:

- . the name of the input file containing the initial conditions
- . a prefix for the output files (incl. the full path)
- . three technical parameter (number of domain grid cells, two refinement criteria)
- . the final time at which AMIGA should exit (in terms of cosmological redshift)

. how often to write a dump file (this is simply an output file with a fixed name and hence the old dump file will be overwritten when writing a new dump file. the dump file just acts as a backup copy of the simulations in case the code crashes...)

- . the required number of output files
- . the times when to write such an output file

NOTE: this information is being asked for by AMIGA interactively and common practice is to write it into a file AMIGA.input and redirect stdin:

```
$ AMIGA < AMIGA.input
```

2. the code further requires a file containing the initial conditions, i.e.
x, v, z, vx, vy, vz, m
for all N particles to be simulated

NOTE: the file with the initial conditions can have various format, i.e. AMIGA's own binary format (the format identical to the output/dump files), GADGET, GADGET-2, TIPSY, ART, and even ASCII.

NOTE: generating these initial conditions is basically a "supercomputer task" on its own and not an integral part of AMIGA :-(

2.3.2 Input data

- How is the input data prepared?

- . AMIGA requires two input files:
 1. initial conditions to be generated by other software
 2. input parameters as specified above:
this file is readily generated/adapted/modified by a shell script

- Where is the input data stored? Describe all central and distributed locations.

- . local file system

- Are file-names known in advance (before the program is started)?

- . filenames should be known and hence could be constructed from meta-data

- Are data locations (directory, server, ...) known in advance?

. same as above

- Describe the different ways data is accessed.

. logfile: fprintf()
. output/dump files: fread() and fwrite()

- Non-file based data access (XML, database, ...) should include description of

. does not apply

- How much data is accessed at each run?

. file with initial conditions varies with respects to problem and scale with the number of particles N: we require positions, velocities and mass for N particles as float variables! That amounts to N x 28 bytes where N ranges from 1e6 to 1e9!!

- Is it possible that a data set/file is accessed multiple times over a short period of time?

. no (only the logfile which is accessed after every time step, however, for state-of-the-art simulations a single step takes hours if not days!)

- How many users are using the same data simultaneously?
Are these users geographically distributed?

. depending on how many people are interested in the physical project, several user from all over the world may wish to post-process the output files using their own analysis software

- Elaborate on the use of metadata related to input data.

. no meta-data yet

2.3.3 Additional Notes

Describe any additional information regarding the input data which has not yet been covered.

2.4 Output

This covers what data products are generated (INTERMEDIATE and FINAL results), where they are generated and how they are handled after the program finished (transferring data or removing it, ...).

2.4.1 Output data

- Where is the output data stored? Describe all centralized or distributed locations.

. local file system

- How is the output data structured?

- . single file
- . data formats: fwrite(x,y,z,vx,vy,vz,m)

- Describe what happens when the program finishes? How are the results used?

- . remains at the output location
- . moved/copied/deleted elsewhere [manually]
 - > depending on user
- . used as input to a subsequent call or to another program
 - > yes

- Describe the different ways data is created/changed.

- . fwrite()

- Non-file based data access (XML, database, ...) should include description of

- . not applicable

- How much data is written by the program at each run?

- . each output file is a copy of the full simulation, i.e. contains x,y,z,vx,vy,vz,m of N particles
- . number of output files depends on project and user requests, respectively

NOTE: some post-processing of the output files is done by AMIGA on-the-fly (if requested by the user and switched on during compile time of AMIGA!) with this feature switched on AMIGA produces some additional output files (predictable names!) containing "skimmed information": AMIGA produces "galaxy catalogues" from the simulation raw data (=particles)

- Describe the parameters which influence the amount of data and number of files/data sets generated.

- . simply the numbers of particles N used for the run
- . and the user specifies how many output files he requires

- Elaborate on the use of metadata related to output data.

- . no applicable

Note, the decision where results are stored is supported by information about the further use or free data storage.

2.4.2 Additional Notes

Describe any additional information regarding the output data which has not yet been covered.

2.5 Information resources

Give a summary of each information resource that is accessed by the program. Include information about data input/output, locations, access methods (XQuery, SQL, ...), security related restrictions, search of metadata (exact key search i.e. "ABC", range queries i.e. "AB*", ...)

2.6 Data Stream Management

Definitions:

Data Stream - intermediate results can be processed by the subsequent processing module before the current module has processed the last element of the input.

- Can single operations be performed on any compute node or do they need special hardware or software?
- Are data exchanged between distributed parts of the application? does this happen at the beginning, during run-time or at the end?
- Are operations compute intensive?

2.7 Resource Security and Access Restriction

Describe all security related information that considers access of resources. User based, Group based, by IP-address/netmask, certificates, nodes/resources within a private network, firewall restrictions, ...

2.8 Additional Information

Give additional information not covered by the sections above.

- How are workflow/pipeline steps interrelated to eachother?
- Is the application executed in several phases where each phase may have different resource requirements or may be executed at a different resource?

. AMIGA dynamically request its memory: at early times the code does not require as much memory as at later times! At early times (=homogeneous and isotropic Universe!) there is no need for the adaptive grids and only the master grid is being used. At later times (gravity tends to clump particles together) a lot of the memory goes into the handling of the adaptive grids...

- How long (avg) does the scenario execute (minutes, hours, days)?

. of order $1E7$ particles take several months to run

- How often will the scenario be executed?

. frequently (about 500 times per simulated universe)

- Are the executions time-critical?

no

3. Future Scenario and AstroGrid-D Usage

Describe the future scenario and envisioned usage of AstroGrid-D as detailed as possible. It is not assumed that the questions can be answered as detailed as in Section 2. Focus on what is expected by the Grid environment and how this new functionality can be used.

Note, there is no special section to describe workflows/pipelines or details about a phased execution of a program. If your scenario is a workflow/pipeline OR your application is executed in several phases, describe EACH step/part covering the sections 3.1 - 3.5. In addition you must describe how these steps/parts are interrelated to eachother (in Section 3.6).

3.0 General goals

- . use more compute resources,
- . provide data to other users,

3.2 Environment

- Are there any constraints due to your participation in other projects or international collaborations?

- . no

3.3 User Interaction

- Which parts should be automated?

- . creation of input file AMIGA.input
- . modifications of technical parameters and Makefile prior to compilation

- Which user interface are you planning to use?

- . WWW portal preferred

- Are you planning to use any standard for application monitoring/steering?

- . logfile access

- Aspects of a Portal / WWW based interface:

- . Which portal features are mandatory/optional (e.g. credential management, job management, job monitoring/steering, data transfer, ...)?

-> job monitoring

. How are user managed? Where is information about users defined / stored?

-> does not apply

- . Which authentication/authorisation methods are needed ?

-> does not apply

. Do you want to access specific data services (web services, databases, etc.) via a portal?

-> no

- . Are there any existing programs, on which the user interface

should be

based OR which should be replaced by the portal?

-> no

. Should there be a central AstroGrid portal OR do you want to set up a portal server for each scenario/application ?

-> central portal preferred

. Does the scenario require any special interfaces OR is it sufficient to use generic interfaces ?

-> generic interfaces should be sufficient

- Aspects of a generic Grid Application Programming API (GAT)

. Which GAT functionality would you like to make use of (eg. job submission, file handling, resource brokering, etc.) ?

-> none

. What programming languages must be supported ? Which platforms ?

-> C

. Which Grid Middleware should be supported (Globus, Unicore, gLite, etc.) ?

-> does not apply

. For specific GAT functionality, which protocols/packages/tools should be supported ?

eg. for job management: clusters with PBS, SGE, Condor

-> does not apply

3.4 Input

- Do you handle input data manually or do you need an automated management of data?

. initial condition files need to be created "manually" using other software packages...

3.5 Output

- Do you handle output data manually or do you need an automated management of data?

. manually

3.6 Additional Information

- How long (avg) does the scenario execute (minutes, hours, days)? Do you aim at a specific speedup?

. months

- How often will the scenario be executed?

. frequently (about 500 times per simulated universe in one run)

- Which restrictions of the current approach (as described in section 2) do you want to overcome?

. memory limitations

4. Bigger Picture for the far future

4.1 Organization of Multiple Runs

Maintain a list of all simulations, to repeat simulations with a different binary, with different input data, to check if a program was already executed with a certain set of parameters/input data,

4.2 Handling relationships between data products

For example, store metadata on how a data product was generated (from which input data, by which program, with which parameters) and how it can be used by others.

4.3 Constructing More Complex Runs

For example, combine existing single programs.

An email exchange clarifying some issues. Initial file (>>, also see use case above) from A. Knebe, questions (>) from T. Roebnitz, answers from A. Knebe:

> Du schreibst, dass das "dumb file" immer wieder überschrieben
> wird und das ein
> Parameter die Anzahl der Ausgabedateien angibt. Das verstehe ich
> nicht ganz.

im Prinzip laeuft eine Simulation so, dass der User sagt, zu welchen
Zeiten

er gerne ein output file haben moechte. Diese output files enthalten die
gesamte Information, die notwendig ist, um die Rechnung an dieser Stelle
fortzufahren.

AMIGA selbst schreibt aber jetzt alle X.Schritte ein output file (X
ist dabei
der Parameter, der die Frequenz der Dump files angibt!): dieses File
hat immer
ein und denselben Namen und ist lediglich eine "Sicherheitskopie" der
Simualtion:
falls AMIGA crasht, kann der User dieses dumpfile verwenden, um die
Rechnung
fortzusetzen.

Diese Sicherheitskopie wurde eingefuehrt, da die Rechnungen teilweise
Monate(!) dauern, und falls das letzte richtige output file zuweit
zurueck liegt, verliert man beim Restart zuviel Zeit...deshalb sollte

des
oeffteren solch ein dump file beschrieben werden...

>> 2.2.2 Monitoring/Steering/Visualization during the run-time of the
>> program
>>
>> - Describe any security measures related to program access for
>> monitoring/steering/visualization.
>> . not specified yet
>
> Was ist der aktuelle Stand? Einschränkung per Zugang/Account der
> jeweiligen
> Maschine?

hmmm, hier ist mir vielleicht die Frage nicht ganz klar?!

AMIGA wird gestartet und es gibt nach dem Start *keine* weitere
Kommunikation
mit dem Programm...es laeuft solange, bis es a) zu Ende ist, b) die
Maschine
abstuerzt, c) AMIGA abstuerzt (eigentlich sollte dies nicht
passieren!), oder
d) der User das "terminateAMIGA" File erzeugt.

Auf das Logfile, welches AMIGA schreibt, kann jeder zugreifen, der dazu
die Leserechte hat...

> Wieviele (serielle) AMIGA-Laeufe werden momentan gleichzeitig
> ausgefuehrt (auf
> moeglicherweise unterschiedlichen Rechnern)?

es ist (fuer eine wissenschaftliche Arbeit) normal, dass man ein paar
"schlecht" aufgeloeoste Simulationen (ca. 2-3 Tage pro Simulation)
laeuft
lasesst. In diesen Daten sucht man dann (per Hand!) nach interessanten
Objekte (=interessante Galaxien) und loest diese in den Anfangswerten
besser auf. Dann laesst man die Simulationen erneut laufen (ca. 3-4
Monate).

Die Anzahl der AMIGA-Laeufe ist somit durch die Anzahl der interessanten
Objekte gegeben.

> Werden momentan Daten aus anderen Quellen als Dateien verwendet?

Nein

> Wird eine Eingabedatei (initial conditions) mehrmals verwendet,
> z.B. mit anderen
> Parametern?

in der Regel nicht. Die Datei mit den Startwerten ist
problemspezifisch und enthaelt

in der Regel die Daten fuer ein interessantes Objekt.

Aber das haengt auch wieder von der Art der Problemstellung ab: man kann auch Parameter-Studien machen, bei denen man ein und dieselbe Startdatei verwendet, aber die technischen Parameter in AMIGA veraendert...

Trotzdem, das Szenario, was ich im Kopf habe ist eher:

jeder serielle AMIGA Job simuliert ein interessantes Objekt, welches vorher aus einer "schlecht aufgeloesten" Simulation herausgearbeitet wurde...

```
>> - How many users are using the same data simultaneously?
>>   Are these users geographically distributed?
>>   . depending on how many people are interested in the physical
>> project,
>>   several user from all over the world may wish to post-process
>> the output
>>   files using their own analysis software
>
> Wie sieht das für die Eingabedaten aus?
```

die schaut sich in der Regel kein Mensch mehr an, wenn die einmal erzeugt wurden...

allerdings sollten die nicht weggeschmissen werden, falls man dann doch evtl. nochmal mit abgeaenderten AMIGA Parametern nachsimulieren moechte...

```
> Evtl. bietet es sich an den Use Case aufzusplitten in (1) die
> parallele
> Ausführung mehrerer AMIGA-Läufe und (2) die Publizierung der
> Daten für
> bestimmte Nutzergruppen.
```

klingt gut: zum einen sollte das GRID verwendet werden, um AMIGA Simulationen durchzufuehren, und zum anderen sollten die fertigen Daten der "Allgemeinheit" zugaenglich gemacht werden.

```
>> 3.3 User Interaction
>>
>> - Which parts should be automated?
>>   . creation of input file AMIGA.input
>
> War das nicht eine äußerst komplexe Angelegenheit? Evtl. müsst
> ihr auch
> einen Workflow spezifizieren:
>
> (i) Erzeugung von AMIGA.input
> (ii) Parameter-Sweep über diese Eingabedatei
```

AMIGA.input ist recht einfach zu erstellen, da dies einfach nur ein paar
Dateinamen und ein paar technische Parameter enthaelt.

Das Problem ist die Datei mit den Startwerten:

AMIGA simuliert die gegenseitige gravitative Wechselwirkung von Mio.
von Teilchen. Und die Startwerte sind jetzt nix weiter als
x,y,z,vx,vy,vz,m fuer die Mio. Teilchen

und diese Startwerte koennen natuerlich nicht einfach beliebig gewaehlt
werden, sondern unterliegen der Physik des (fruehen) Universums! Deshalb
ist die Erzeugung dieser Datei ein eigenstaendiges Problem, fuer das
wiederrum andere Software verwendet wird...

```
>> - Do you handle input data manually or do you need an automated
>> management of
>>   data?
>>   . initial condition files need to be created "manually" using
>> other software packages...
>
>   Ok, hier geht es mehr um die Frage, ob das Programm selbst die
> Daten zum
> Ort der Berechnung kopiert oder ob dies automatisch passieren soll.
```

das Programm kann die Daten lediglich von irgendwo im NFS lesen. Wenn
dort
nicht vorhanden, sollten die Daten automatisch kopiert werden..

```
>> - How often will the scenario be executed?
>>   . once
>
>   Pro Eingabedatei oder insgesamt?
```

eine Simulation = ein interessantes Objekt = einmalige Ausfuehrung

```
>
>> - Which restrictions of the current approach (as described in
>> section 2)
>>   do you want to overcome?
>>   . memory limitations
>
>   Durch Parallelisierung oder durch Verwendung von Ressourcen mit
> größerem
> Speicher?
```

wir arbeiten derzeit an einer Parallelisierung von AMIGA um diese
Limitierung
zu umgehen: Verteilung des benoetigten Speichers auf mehrere CPU
durch Verteilung
der N Teilchen auf die CPU's. Aber auch eine Einzel-CPU mit genuegend
Speicher
wuerde dem Problem nicht mehr gerecht werden, da die Rechenzeiten
fuer noch

groessere N einfach zu lange sind (fuer einige Mio. sind es ja jetzt schon einige Monate, und zum besseren Verstaendnis der Strukturen im Universum werden noch viiiiel, viiiiel mehr Teilchen benoetigt!!!)

Use Case written for the 1st Call (15.11.05)

AIP: Use Cases

=====

aknebe@aip.de, Alexander Knebe, 0331-7499535

=====

Programmname: AMIGA

Anwendungsbereich: Studie der Umlaufbahnen von Satelliten Galaxien in kosmologischen Dunkle Materie Halos

Code-Art: Loesung der Poisson-Gleichung mittels adaptivem Gitter-Verfahren

Code: C

Libs: --

Parallellisierbar: in Arbeit

Memory Requirements: 1-16 GB

OS: keine speziellen Anforderungen (ANSI-C)

Local diskpace: >=50 GB

Einsatz/Workflow:

Input:

1. bis zu 3GB grosse (binaer) Files mit Anfangswerten (Positionen und Geschwindigkeiten von bis zu 150mio. Teilchen)
2. Parameterfile (manuell editieren oder via Script erzeugen)

Run:

ASCII Logfile mit speziellen Kenngrößen

Output:

1. Zeitfolge von Simulationsschritten (=restart-files), bis zu 3GB pro Ausgabefile (bei restart muss Inpufile entsprechend angepasst werden)
2. Galaxienkataloge (=Analyse der Simulationsschritte), bis zu 500MB pro Analyse

Verarbeitung:

IDL-Routinen zur Visualisierung und Analyse der Galaxienkataloge

Anforderungen: interaktiver Zugriff auf Logfile

laengerfristig:

1. remote Zugriff/Visualisierung von (binaer) Daten (mit IDL)
2. automatische Parameter-Erzeugung auf Basis vorangegangener und laufender runs (Job-Steering)

Einsatzbereich: Taskfarming (keine Kommunikation zwischen den Runs):

1. Nach-Simulierung (mit erhoehter Aufloesung) diverser Dunkle Materie Halos (ein Halo pro CPU)
2. Erstellung von Galaxienkatalogen anderer kosmologischer Simulationen (ein Katalog pro CPU) (es liegen Simulationen mit bis zu 500 Ausgabefiles pro Simulation vor...)

Einsatz Grid: bessere Resource-Ausnutzung

Notwendige Grid-Anpassungen:

- Wrapper fuer Automatisierung von Input-Parameterfiles fuer
- 1. Nach-Simulierung (eigenstaendige Simulationen)
- 2. Analyse bestehender Daten

Interface: Sollte Start von Scripten zum Jobmonitoring gestatten, (Portal?)

Bemerkung Angelika: Hier hat man ein großes Inputfile von Teilchen. Es gibt zwei Parallelisierungsmoeglichkeiten: Entweder man hat einen Masterprozess, der die Berechnung steuert und im Laufe der Berechnung Teile zur Weiterberechnung auslagert, mit Datentransfer, je nach Berechnungsverlauf (es bildet sich eine Galaxie heraus, die getrennt weiterberechnet werden kann. Oder man verteilt das Inputfile gleich auf mehrere Knoten, lässt diese verteilt rechnen und dann wird hier aber im Laufe der Berechnungen Koordination zwischen einzelnen Knoten notwendig werden, die auch großen Datentransfer beinhaltet.

Antworten von Alexander Knebe auf Email von Thomas Röblitz

- > - Ausgabedateien:
- > . Wo werden sie momentan gespeichert? In einer Partition auf dem
- > Master-Knoten der Anwendung (bei Benutzung von MPI o.Ä.)? In
- > einem NFS-Verzeichnis?

Im Input-File fuer AMIGA wird der Pfad fuer die Ausgabefiles vom User angegeben. Da AMIGA ein serieller Code ist, stellt sich nicht die Frage nach Master oder Slave...

Diese Ausgabedateien beinhalten die Positionen, Geschwindigkeiten und Massen von allen Teilchen. Die Groesse der Files ist somit direkt proportional zur Anzahl der simulierten Teilchen (einfach $7 \times \text{float} \times N$). N bewegt sich im Bereich von 128^3 (56MB/File) bis 1024^3 (28GB/File).

Pro Simulation werden von ca. 10 bis hin zu hunderten solcher Output-Files geschrieben (haengt von der jeweiligen Art und "Fragestellung" der Simulation ab). Aber man kann so im Mittel davon ausgehen, dass eine "vernuenftige" Simulation, mit der man Wissenschaft betreiben kann, an die 10-20GB produziert.

Da AMIGA ein serieller Code ist, befasse ich mich in erster Linie mit Parameter-Studien bzw. versuche statistische Analysen zu betreiben, so dass ich in der Groessenordnung 5-20 vergleichbarer Simulationen benoetige.

- > . Wenn Dateien auf einer lokalen Partition (Master-Knoten) erzeugt
- > werden, was passiert mit ihnen nach Programmende? Wohin werden
- > sie kopiert?

bisher werden die Roh-Daten dort weiter analysiert, wo sie auch geschrieben worden sind...

> . Sind die Namen der Ausgabedateien vor der Ausführung bekannt?

ja, da sie vom User frei wählbar sind...

allerdings wird dem sog. Prefix, welches der User wählt, der Integrationsschritt nachgestellt...und der ist auf Grund der adaptiven Schrittanpassung an das Problem nicht eindeutig vorhersehbar.

> - Programmausführung:

> . Alle Programme haben keine speziellen Anforderungen an das Betriebssystem. In welchen Umgebungen werden sie eingesetzt?
> Windows/Linux/Solaris/..., Workstation/Cluster/...

bisher wurde AMIGA auf allen mir bekannten Unix-Derivaten eingesetzt (inkl. Mac OS X) und es läuft auch auf Windows Rechnern ohne Probleme...

AMIGA benötigt lediglich einen C-Compiler...

> Ist es vorstellbar, dass die Programme erst zu Beginn der Ausführung kompiliert werden?

ja

> Gibt es vorkompilierte Binaries für unterschiedliche Umgebungen?

auch möglich...

aber während der Compilations-Phase können einige (technische!) Parameter gesetzt werden, die evtl. für die Parameter-Studie von Relevanz sein könnten...

> . Ist bekannt wie lange eine Ausführung dauert? Sind die Anforderungen an Speicher und Disk vor dem Programmstart bekannt (in Abhängigkeit der Parameter)?

hier können nur grobe Abschätzungen basierend auf Erfahrungen gemacht werden: es gibt keine einfache Formel für die Laufzeit bzw. den Gesamt-Speicherbedarf.

> . Wie wird der interaktive Zugriff auf das/ein Logfile durchgeführt?
> Wird der gesamte Inhalt/ein Teilinhalt einmal/mehrmals gelesen?

wenn AMIGA einmal seine Startup-Phase überwunden hat (Einlesen der Startdaten etc.), dann werden sämtliche Information in ein Logfile geschrieben (kein Output zu stdout oder stderr!).

Dieses logfile dient in erster Linie der Abschätzung, wie lange der Code noch laufen wird und wie akkurat die Ergebnisse momentan sind. Es findet keine "Visualisierung des Logfiles" statt!

Allerdings werden im Laufe der Simulation die Output Files (s.o.) geschrieben, wobei wiederum unklar ist, wann genau solch ein File erzeugt wird. Das haengt in erster Linie von der "Schnelligkeit" der Rechnung ab. (erlaeuterndes Beispiel: alle 10 Integrations-Schritte soll ein Output-File erzeugt werden -> aber die Simulation wird im Laufe der Zeit immer langsamer, so dass die "Output-Frequenz" zum Ende hin abnimmt...)

- > . Wie erfolgt der 'remote' (von wo aus?) Zugriff für die
- > Visualisierung?
- > Wird auf temporäre Dateien oder über eine Schnittstelle des
- > Programmes
- > auf die Daten zugegriffen? Werden alle Daten oder nur Teile
- > gelesen?

die Visualisierung erfolgt momentan durch den Transfer von Analyse-Files auf lokale Rechner: die Rohdaten (=Output-Files) werden vor Ort mit diversen Analyse-Programme ausgewertet, und deren Output wird dann per scp auf lokale Rechner transferiert.

- > . Welche Metadaten einer Programmausführung sollen gespeichert
- > werden,
- > um sie für die Erzeugung neuer Parametersätze zu verwenden?

Die Output-Files selbst enthalten einen Header, der saemtliche relevanten Meta-Daten der Simulation beinhaltet.

- > - Interface: Starten von Skripten zum Jobmonitoring betrifft den
- > inter-
- > aktiven Zugriff auf Logfiles?

nein

- > - Eingabedatei:
- > . Gibt es eine begrenzte Menge von Eingabedateien oder werden diese
- > für jede Ausführung generiert?

die Eingabe-Dateien sind *sehr* knapp gehalten und beinhalten die folgenden Informationen:

- wie heisst das File mit den Startwerten
- was fuer ein Prefix soll fuer die Output-Files (inkl. Pfad) genommen werden
- 4 technische Parameter
- wieviele Output-Files sollen geschrieben werden
- wann sollen diese Output-Files geschrieben werden

- > . Wird eine Eingabedatei mehrmals verwendet (Parameterstudie)?

in erster Linie bedeutet eine Parameter-Studie, dass man

unterschiedliche
Startwerte mit ein und demselben AMIGA laufen laesst. Somit sind auch
die
Eingabe-Dateien identisch, bis auf den Namen des Files mit den
Startwerten!

> - Wie werden die berechneten Galaxienkataloge verwendet/aufbewahrt?

die Galaxienkataloge werden (wenn gewuenscht!) schon waehrend der
Simulation von AMIGA erzeugt und werden dann per scp auf einen lokalen
Rechner zur weiteren Analyse transferiert.

> - Wie wird das Programm bisher verwendet? Workstations, Cluster, ...

Die Analyse der Galaxienkataloge erfolgt lokal auf Workstations,
die Simulationen auf Clustern (eine Simulation pro CPU)

Eine Anmerkung am Rande: die Erzeugung der Startwerte ist ein nicht-
triviales
Problem und benoetigt widerrum den Einsatz von Grossrechnern! Alle meine
Beschreibungen gehen momentan davon aus, dass man von irgendwo die
Startwerte erhalten hat und sich um deren Generierung keine Sorgen
machen muss...AMIGA selbst kann keine Startwerte erzeugen!! AMIGA kann
sie nur "weiterverarbeiten"...