

A New Parallelization Approach for Sequential Simulation

H.S. Vargas, H. Caetano and H. Mata-Lima

Abstract The procedure for spatial sequential simulation – bi-point or multi-point stochastic simulation – of any type of variable starts with the definition of a random path which the simulation should follow in order to generate a structured image of a given attribute. One problem of these algorithms is related to the effort a single processor is required to undertake, especially when applying them to very large grids of nodes.

With the advent of parallel computing and multi-core processors (or multiple execution cores), which are expected to drive a new era of performance and flexibility, providing platforms that can better handle escalating workloads and rapidly evolving usage models, it becomes clear that a scalable parallelization scheme should be developed to allow the usage of such processors to allow for considerable reduction in time spent performing simulations, with clear advantages when used with clusters of multi-processor (or multiple execution core) nodes.

The general idea is to partition the universe in a given number of sections, equal in number to double the number of processors or execution cores, in such a way that the locations to be concurrently simulated are sufficiently apart to be outside search range or multi-point template range. This is only applicable in cases where at least one of the dimensions of the area to be simulated is greater than the chosen range in that direction, which is admitted to be true for cases where parallelization is valuable, particularly for very large fine scale models.

The number of sections, or regions, at which the volume will be segmented is given by an optimization procedure that maximizes the size of each region and minimizes the number of nodes to be sequentially simulated, based on the number of available processors or execution cores.

The results of the proposed parallel simulation method were checked in order to evaluate if they succeeded to reproduce the spatial continuity and spatial patterns of the phenomenon and its distribution function.

H.S. Vargas

CMRP – Centre for Modelling Petroleum Reservoirs, Mining and Georesources Department,
Instituto Superior Técnico, Av. Rovisco Pais, 1049-001, Lisboa, Portugal
e-mail: hugo.vargas@total.com

1 Introduction

In recent years, stochastic simulation has increasingly become an indispensable tool for generating equal probability images of a set of random variables with joint probability distribution law, as shown in many references (Goovaerts, 1997, Deutsch and Journel, 1998, Soares, 2006

In earth sciences, the sequential simulation algorithms developed up-to-date, either based on **bi-point** statistics (Sequential Gaussian Simulation – SGS –, Sequential Indicator Simulation – SIS –, Direct Sequential Simulation – DSS –), or **multi-point** statistics (SIMPAT, SNESIM, filterSIM), share a common constraint, imposed by the conditioning data: the simulation of a given node $Z(x_m)$, $1 < m < n$, imposes the knowledge of all the previously simulated values $Z(x_l)$, $l < m$, that became conditioning data and all the experimental data. Therefore, the crucial point of the **sequential simulation** method is knowing the N cdf: $\text{prob}\{Z(x_1) < z|(n)\}$, $\text{prob}\{Z(x_2) < z|(n+1)\}$, $\text{prob}\{Z(x_3) < z|(n+2)\}$, ..., and $\text{prob}\{Z(x_N) < z|(n+N-1)\}$. This is usually a problem since as the number of simulated nodes/conditioning data increases, the estimator of the probability of z_0 belonging to X becomes unmanageable and inaccurate (Gomez-Hernandez, J., Journel, A., 1993). An approximation to this can be obtained through selecting a limited number of conditioning data $\{n_1\} \subset \{n\}$ with $n_1 \ll n$ so that we have $\text{prob}\{Z(x_0) < z|(n_1)\} \approx \text{prob}\{Z(x_0) < z|(n)\}$.

Reduction in time spent performing simulations is a challenge in numerous branches of sciences, particularly in earth sciences in which simulations are related to huge models with millions of cells. In order to mitigate this trouble Dimitrakopoulos, R. (2004) proposed a combination of scales to simulate large grids. But a real parallelization algorithm was yet to be developed.

Reducing simulation effort can be achieved either by sharing global simulation effort or sharing individual simulation effort. In this work we propose a method which allows parallelization of simulation or estimation procedures by sharing the effort of each individual simulation with a great number of processors, exploring the capabilities of newer multi-core processors.

2 Proposed Approach

Using a sequential simulation procedure to simulate a dependent variable in a n -dimensional Cartesian space V with a finite number of nodes N , at location $x_1 - Z(x_1) -$, the experimental data points and/or previously simulated nodes to be included as simulation constraints are those which fall within variogram range, search range or multi-point template (referred to as range). Any experimental data points or previously simulated nodes outside this range will be ignored. In practice, when working with very large fine-scale grids with fairly small ranges (when compared with image dimensions), if we have spare processing power, it is possible to simultaneously simulate a second dependent variable $Z(x_2)$ (or third, or

nth) as long as their ranges do not intersect. There are many ways to ensure this condition; one of them is defining a number of regions of V , according to certain rules, involving the following concepts: Region Definition, Region Association and Node Shuffling (coordinate pair shuffling). We assume that classical ways of defining unique random paths will suffice for our path choosing step of algorithm implementation.

2.1 Region Definition

Considering we have a number of processors p , with shared memory, we are able to simultaneously simulate the same number of dependent variables if we define an optimum number of lists, l , of dependent variable locations (grid nodes) to be simulated for each processor, with each list representing the number of regions in which the initial space will be divided. These lists are ordered sets of nodes of a regular grid to be simulated.

If $V = v_1 \times v_2 \times \dots \times v_n$, with n being the number of dimensions, we also have a set of ranges R , one for each dimension of space, $R = \{r_1, r_2, \dots, r_n\}$, taken from the multi-point template or defined by search range.

To verify if we can define regions of space V , the total number of cells to be simulated in one of its dimensions has to allow being divided by its range with a result greater than the number of lists:

$$v_\alpha / r_\alpha \geq l, \quad \alpha \in \{1, \dots, n\}.$$

The number of regions of space is defined by l .

2.2 Region Association

Being that V is composed by an ordered set of contiguous regions $V = \bigcup_{l=1}^n w_l$, with l defined above, these have to be associated in proper subsets such as duplets, triplets, ..., p -tuplets, which will then be added to a list (ordered set).

We have to ensure that within any p -tuple there are only regions which are sufficiently apart to be simultaneously simulated, which, in this context, means being non-contiguous. In this way, a list of p -tuplets, L_p , is defined as an $n(V)/p$ size list of non-repeating proper subsets of p non-contiguous non-repeating regions.

2.3 Node Shuffling

Each region that composes a p -tuple represents a set of Cartesian coordinates (regular grid nodes) which can be sorted according to any criteria. Each possible sorting represents a path for the simulation of the dependent variable within that

list. Consequently, at this stage, there is a list composed of a p -tuple of lists of nodes that can be simultaneously simulated. This list of lists has to be transformed in a unique shuffled list of p -tuples of nodes to prevent the complete simulation of any given region before starting the simulation of the following. This can be done by removing a randomly chosen coordinate p -tuple of coordinates from the list, of initial size a , inserting it into another list and executing this until the second list has reached size a , or, in other words, until the first list is empty.

3 Example

This parallelization approach was tested with a sequential simulation procedure with elevation data from Portugal, on a P4 machine with 2 CPUs. Ninety images of the attribute in a testing image is $121 \times 2440 = 295\,240$ pixels were generated in three groups in the following way: thirty images using two CPUs, thirty images using a single CPU and a path choice algorithm different than the one implemented by the methodology, and thirty simulations using a single CPU and the same path choice algorithm as the two CPU version. The idea is to compare the first two groups of images in order to evaluate if the first group of simulations could be considered as

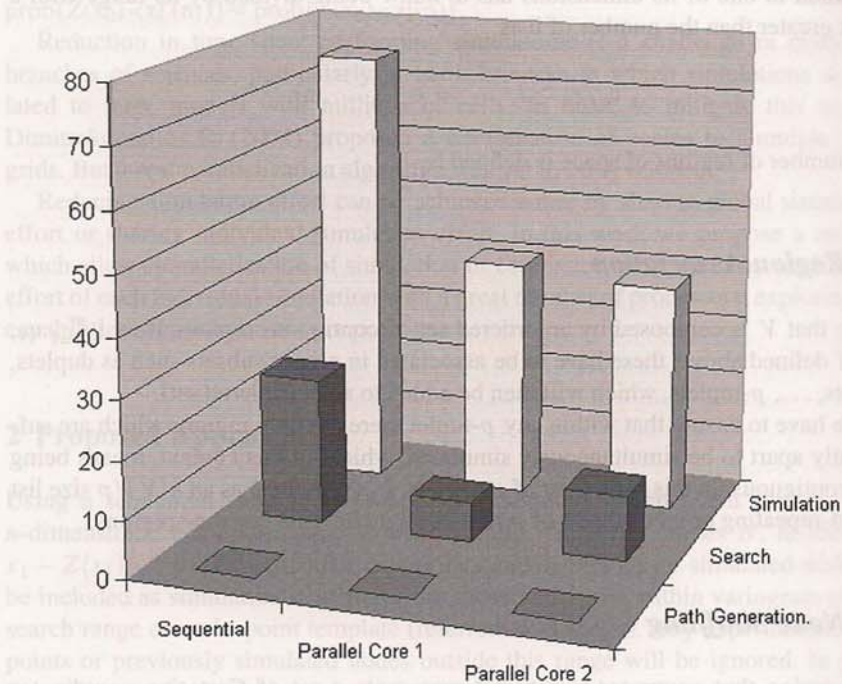


Fig. 1 Graph showing the percentage of time a single-core processor and the dual processor machine used to simulate this example take to execute the simulation

the result of a similar process and if their statistics match. The last group of simulations has the purpose of demonstrating that by using the same well-defined list of simulation nodes with 1 or two CPUs yields the same result. Figure 1 summarizes the results concerning time saving.

The simulation procedure was divided in three steps: Path generation, search for neighboring simulated nodes (or hard data) and simulation of the attribute. Figure 1, on the left, shows the time it takes a single processor to simulate a single image (the sum of path generation, search and simulation). This is our reference. On the right, figure 1 shows the time charge of the each of the two processors relative to the single processor procedure. It is clear that the charge has effectively been divided by the two processors. There is, however, a charge that was not, in this case, divided by the number of processors: the path generation procedure. But it is clear that its effect is negligible in this example.

Figures 2a) and 2b) show a result obtained by using the parallel sequential procedure and the purely sequential procedure, with the same seed value. Figure 1c shows the result of simulating the properties of the attribute using the same path as the one used for the parallel procedure with a pure sequential simulation.

We believe that figure 2a) and 2b) show that the images are the result of a similar process, which figure 2c) reinforces, as does their statistical behavior, shown in fig. 3 and table 1.

In what refers honoring the variograms, figures 4a) through 4h) show the hard data variogram and the variogram of the every type of simulation involved in the testing.

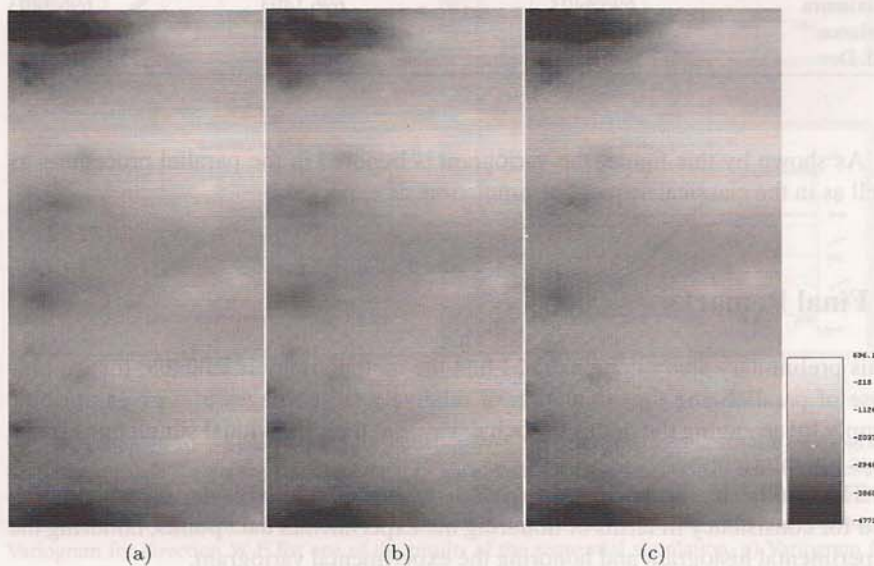


Fig. 2 a) Simulation result 1 from the group of parallel simulations; b) Result 1 from the group of pure sequential simulations 1; c) Result 1 from the group pure sequential simulations with the same path as the parallel sequential simulations. Elevation values in meters (m)

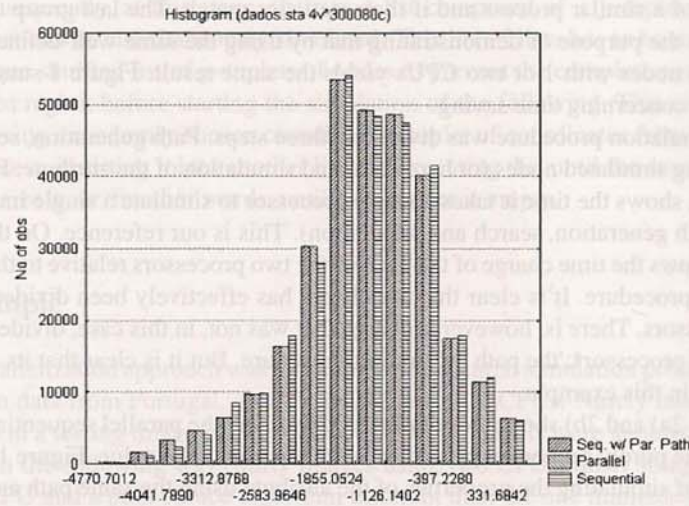


Fig. 3 Global histograms for the images of the attribute shown in Fig. 1

Table 1 Basic statistics of the images shown in Fig. 1

	Seq. w/ Par. Path	Parallel	Sequential
Valid N	300080	300080	300080
Mean	-1360.40	-1360.40	-1353.23
Minimum	-4770.70	-4770.70	-4770.70
Maximum	696.1403	696.1403	696.1403
Variance	785100	785100	770029
Std. Dev	886.059	886.059	877.513

As shown by this figure, the variogram is honored in the parallel procedure, as well as in the classical sequential simulation, as expected.

4 Final Remarks

This preliminary study demonstrates that the methodology is valuable for the purpose of parallelizing the simulation of relatively large images of a given attribute simply by arranging the nodes in such a way that their individual simulations range does not interfere.

The results obtained using this parallelization methodology were tested and verified for consistency in terms of honoring the experimental data points, honoring the experimental histogram and honoring the experimental variogram.

In practice this approach has proven to allow saving time in what concerns, exclusively, the simulation of the dependent variable, reducing time-consumption by a ratio which depends on the number of processors available. The methodology

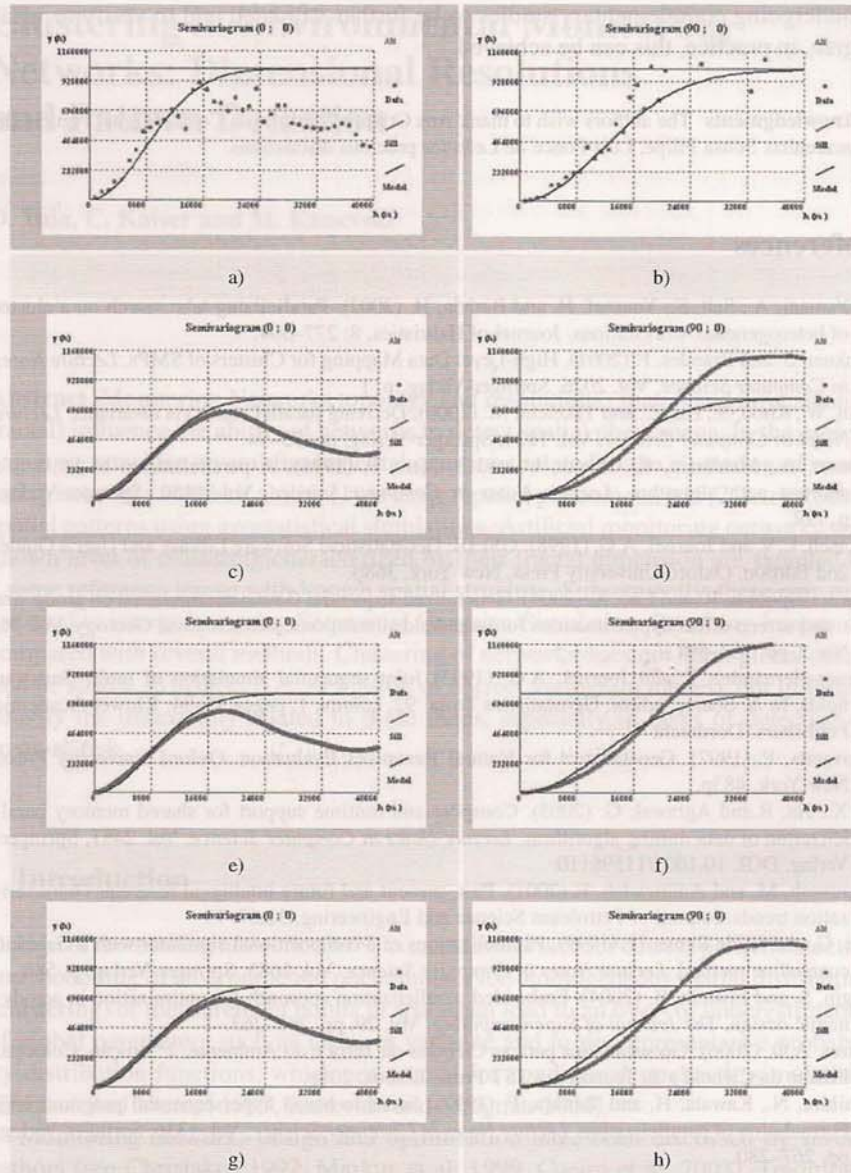


Fig. 4 a) through h) – a) Variogram for direction N-S for the experimental data, b) Variogram for direction W-E for the experimental data, c) Variogram for direction N-S for one of the results of the parallel simulation, d) Variogram for direction W-E for one of the results of the parallel simulation, e) Variogram for direction N-S for one of the results of the sequential simulation, f) Variogram for direction W-E for one of the results of the sequential simulation, g) Variogram for direction N-S for one of the results of the sequential simulation following a simulation path defined by the parallel procedure., h) Variogram for direction W-E for one of the results of the sequential simulation following a simulation path defined by the parallel procedure.

is undergoing complementary studies to be further validated and to verify to what degree, in practice, this can be achieved.

Acknowledgments The authors wish to thank Ana Cristina Marinho da Costa, Miguel Figueiredo Mascarenhas Sousa Filipe, Luis Ponce de Leão for precious discussions.

References

- Al-Yamani, A., Sait, S., Youssef, H. and Barada, H. (2002). Parallelizing tabu search on a cluster of heterogeneous workstations. *Journal of Heuristics*, 8: 277–304.
- Benkner, S. and Brandes, F. (2001). High-Level Data Mapping for Clusters of SMPs. *Lecture Notes in Computer Science*, Vol. 2026, Springer-Verlag, p. 1
- Chin, W., Khoo, S., Hu, Z., and Takeichi, M. (2000). Deriving parallel codes via invariants. *Lecture Notes in Computer Science*, Vol. 1824, Springer-Verlag, pp. 75–94.
- Crauser, A., Mehlhorn, K., Meyer, U. and Sanders, P. (1998). A parallelization of Dijkstra's shortest path algorithm. *Lecture Notes in Computer Science*, Vol. 1450, Springer-Verlag, p. 722.
- Deutsch, C.V. and Journal, A.G. (1998). *GSLIB: Geostatistical Software Library and User's Guide*, 2nd Edition, Oxford University Press, New York, 368p.
- Dimitrakopoulos, R. and Luo, X. (2004). Generalized sequential Gaussian simulation on group size ν and screen-effect approximations for large field simulations. *Mathematical Geology*, Vol. 36, No. 5, pp. 567–591.
- Gómez-Hernández, J. and Journel, A.G. (1993) Joint sequential simulation of multi Gaussian fields. In A. Soares, editor, *Geostatistics Troia '92*, volume 1, pages 85–94. Kluwer Academic Publishers. Dordrecht
- Goovaerts, P. (1997). *Geostatistics for Natural Resources Evaluation*. Oxford University Press, New York. 483p.
- Li, X., Jin, R. and Agrawal, G. (2005). Compiler and runtime support for shared memory parallelization of data mining algorithms. *Lecture Notes in Computer Science*, Vol. 2481, Springer-Verlag. DOI: 10.1007/11596110
- Nikravesh, M. and Aminzadeh, F. (2001). Past, present and future intelligent reservoir characterization trends. *Journal of Petroleum Science and Engineering*, 31:67–79
- Øye, G. and Hilde Reme, H. (1999). Parallelizations of a compositional simulator with a Galerkin coarse/fine method. *Lecture Notes in Computer Science*, Vol. 1685, Springer-Verlag, p. 586.
- Peigin, S. and Epstein, B. (2004). Embedded parallelization approach for optimization in aerodynamic design. *The Journal of Supercomputing*, Vol. 29, pp. 243–263.
- Soares, A.O. (2006). *Geostatística para as Ciências da Terra e do Ambiente*. 2ª Edição, Coleção Ensino da Ciência e da Tecnologia, IST Press, Lisboa, 206p.
- Uchihira, N., Kawata, H. and Tamura, F. (1997). Scenario-based hypersequential programming: Formulation of parallelization. *Lecture Notes in Computer Science*, Vol. 1336, Springer-Verlag, pp. 267–280.