NOTES AND CORRESPONDENCE

Suitability of GCM Physics for Execution on SIMD Parallel Computers

By Leon Rotstayn

CSIRO Division of Atmospheric Research, Mordialloc, Australia, 3195

Rhys Francis, David Abramson

CSIRO Division of Information, Technology, Carlton, Australia, 3053

and

Martin Dix

CSIRO Division of Atmospheric Research, Mordialloc, Australia, 3195
(Manuscript received 29 September 1992, in revised form 24 December 1992)

Abstract

A possible path to the teraflop performance that will be required by the next generation of GCMs (general circulation models) is the SIMD (Single Instruction stream, Multiple Data stream) massively parallel computer architecture. A critical consideration in moving a model to a SIMD architecture is the efficiency of the model's physical parameterizations on this type of machine. We have analysed the physics in a production GCM to evaluate its potential for efficient SIMD execution, on the assumption that each processor of the computer is allocated to a single vertical column of the model. This paper summarizes the results obtained for the various physics routines and compares these with the efficiencies obtained under the assumption of MIMD (Multiple Instruction stream, Multiple Data stream) execution. Overall, we found a performance penalty of only 15 % to 20 % for SIMD compared to MIMD execution. This is a very acceptable result, which suggests that SIMD computers should not be rejected for climate modelling (or numerical weather prediction) applications.

1. Introduction

Climate modelling is such a heavy consumer of computer resources that long-term climate prediction is considered to be one of the "grand challenges" of high performance computing. It is thought likely that the teraflop (10^{12} floating point operations per second) performance that will be required by the next generation of general circulation models (GCMs) will only be achieved via massively parallel computers (U.S. Department of Energy, 1990). A possible path to teraflop performance is the SIMD (Single Instruction stream, Multiple Data stream) massively parallel architecture. In a SIMD computer, all the processors execute the *same* instruction simultaneously, applying it to different input data. Compared to MIMD (Multiple Instruction stream, Multiple Data stream) parallel computers, SIMD computers are simpler to program, and have a simpler structure, which allows them to be more highly parallel. On the other hand, they offer less flexibility, due to the restriction that all processors must execute the same instruction simultaneously. Kauranne (1990) has described and compared the different types of massively parallel architectures which are available.

An obvious way to run a GCM on a SIMD computer would be to assign one processing element (PE) of the computer to each vertical column in the model. This configuration allows the largest possible number of PEs to be utilized, without attempt-

ing to also parallelize over the vertical dimension. Vertical parallelization is difficult, as the algorithms used for the physics in current GCMs contain data dependencies in the vertical or require extensive vertical communication. Furthermore, previous results (Francis *et al.*, 1992) have suggested that such a three-dimensional parallelization would not achieve acceptable efficiency on SIMD architectures. On the other hand, the physics calculations in each vertical column are independent of each other and require essentially no communication between columns, making a two-dimensional parallelization far more appealing. However, for SIMD machines in particular there is still the critical question of the efficiency of the physics routines, which generally contain many segments of code which are executed only at subsets of the gridpoints[1]. It is this characteristic of the physical parameterizations which limits the maximum performance that may be achieved by the physics routines on SIMD machines, given that the communications overhead in the physics routines is negligible. Swarztrauber and Sato (1990) and Sato (1992) have considered the SIMD execution of finite difference and spectral shallow water models and Tuccillo (1990) successfully implemented a finite difference primitive equations model on a SIMD machine. However, the models used in these studies did not include physical parameterizations, and the efficiency of the physics routines is still unknown. A low efficiency would result in a large difference between peak and delivered performance for climate or numerical weather prediction (NWP) models on SIMD machines, possibly rendering SIMD execution ineffective. Furthermore, a low SIMD efficiency could not be easily addressed by changing the algorithms used for the physical parameterizations, because it would result from the spatial variability of the physical processes. This is in contrast to the dynamics, where, for example, if the spectral methods used by many GCMs should prove to be unsuitable for massively parallel execution due to the communications overhead, there are other forms of discretization which may be employed (*e.g.* Chern and Foster, 1992).

In view of the large amount of recoding that would be required to move such models to SIMD architectures, it is worthwhile to investigate the likely performance of the physics routines before embarking on such a task. We have analyzed the physics in a production GCM to evaluate its potential for efficient SIMD execution. The model is a version of CSIRO9, the CSIRO nine level R21 spectral model, which is currently being used in the study of climate change. CSIRO9 is an improved form of the CSIRO four level model which has been used extensively in the study of climate change and in drought prediction studies (*e.g.* Hunt and Gordon, 1989; Gordon and Hunt, 1991). The physical parameterizations include a version of the Fels-Schwartzkopf radiation scheme with three layers of diagnostic clouds, two soil moisture layers, gravity-wave drag, a mass-flux cumulus convection scheme, shallow convection and a stability dependent boundary layer. The model timestep is 30 minutes at R21 resolution. The grid used for the physics at this resolution consists of 64 evenly spaced longitudes and 56 Gaussian latitudes, giving a total of 3584 points. Thus the mapping of gridpoints to PEs considered here allows the use of up to 3584 PEs even for this low resolution GCM—higher resolution models would of course allow the use of larger machines.

In this paper we outline the method used to analyze the physics code, summarize the results obtained and compare these with the efficiency of the physics under MIMD execution. We then briefly discuss the implications of these results for the future of SIMD climate modelling.

2. Analysis method

The method used to analyze the model code can be applied to any Fortran code suitable for parallelization and is described in detail by Francis *et al.* (1992). Essentially, the procedure involved the following steps:

1. Scan the model Fortran code with a program which inserts calls to a library of routines which accumulate the flops (floating point operations) at each gridpoint in each basic block[2] of code.
2. Compile the processed Fortran and run the model for one timestep, saving the flop counts to a data file.
3. Using another program, analyze the data on the assumption of either SIMD or MIMD execution and a particular configuration of PEs (in this case one PE per vertical column).

Consider the (SIMD or MIMD) execution of a code segment consisting of N_b basic blocks on a chosen notional configuration of N_p PEs. The **serial flops** for this code segment is obtained by summing the serial flops performed in each basic block by each PE, *i.e.*

$$F = \sum_{j=1}^{N_b} \sum_{i=1}^{N_p} F_{ij} \tag{1}$$

where F_{ij} is the number of serial flops performed by PE i in basic block j. The **SIMD flops** for the code segment is defined as

[1] If the model is, for example, melting sea ice at one gridpoint, then every other PE will be idle until this calculation is completed, due to the single instruction requirement.

[2] A *basic block* is a compiler term which refers to a segment of code which is not broken up by any statements which affect flow control.

$$S = \sum_{j=1}^{N_b} (\max_{i=1}^{N_p} F_{ij}). \quad (2)$$

Note that if a given basic block contains only "in-line" Fortran with no calls to the mathematics library, the code in the block is either executed or not executed on each PE, resulting in only two possible values of F_{ij} (0 or the number of flops). However, where the Fortran calls the mathematics library, different executions could take different paths (depending on the arguments passed to the library) and hence accumulate different flops. We therefore take the maximum number of flops as the effective SIMD work in Eq. (2). The **MIMD flops** for the code segment is the maximum number of flops executed by any of the PEs, *i.e.*

$$M = \max_{i=1}^{N_p} \left(\sum_{j=1}^{N_b} F_{ij} \right). \quad (3)$$

The **SIMD efficiency** of the code segment is the percentage of PEs that would be in use (on average) under SIMD execution, *i.e.*

$$E_s = \frac{F}{N_p S} \times 100\,\%. \quad (4)$$

The **MIMD efficiency**,

$$E_m = \frac{F}{N_p M} \times 100\,\% \quad (5)$$

is most meaningful when evaluated for the physics as a whole, since in practice it would not be necessary to synchronize at the end of each routine. However we have also calculated MIMD efficiencies for each routine separately, as these provide an upper bound on the SIMD efficiencies achievable and may be regarded as a baseline with which to compare the SIMD results.

Two key assumptions underlying the method used to analyze the code are:

1. The flops executed in the program are necessary to the result. For example, the version of the Fels-Schwarzkopf radiation scheme which is used operationally in CSIRO9 performs the short wave calculation at all gridpoints, even those which are in darkness, in order to vectorize efficiently. For the purpose of this analysis, we reverted to an older version of the scheme which does not share this feature.

2. Performance is determined by flops. In particular, communication costs are ignored. Note, however, that communication costs would be negligible for the physics in CSIRO9, and in other GCMs of which we are aware. The SIMD and MIMD efficiencies presented here provide an upper bound on the speedup that may be achieved by the GCM physics on each type of architecture. Thus, they may be thought of as generalizations of Amdahl's Law[3], which account for the fact that even in parallel segments of code, not all PE's are necessarily active, *i.e.* parallel code may not be perfectly parallel.

The results presented here were obtained on a Sun workstation. The model was also run for one timestep on a Cray Y-MP, giving flop counts which differed slightly, because the flop counting hardware on the Y-MP counts integer operations as flops, counts divisions as four flops, and because the mathematics libraries on the two machines are coded differently. Despite this, the efficiency figures obtained on the two machines were nearly identical. We present results only for a timestep performed under UTC 0000, February 1 conditions. Earlier runs performed under January 1 and July 1 conditions and for different times of day showed no significant differences in the overall results, so we have taken the results presented here to be representative of a much longer model run.

3. Results and discussion

The model physics essentially consists of the following subroutine calls in the order given:

- *radin*: Interface to the following physics routines.
- *surfset*: Set surface type, albedo, roughness length etc.
- *qsgam*: Calculate saturation mixing ratio and relative humidity.
- *sflux*: Calculate surface fluxes of heat, momentum and moisture.
- *radfs*: Diagnose clouds. Calculate radiative heating rates, based on full diurnal and annual cycles.
- *surfupa*: Update the soil temperature and moisture fields using the surface fluxes.
- *vertmix*: Apply vertical mixing and shallow convection.
- *gwdrag*: Calculate gravity wave drag.
- *rainda*: Calculate large scale precipitation.
- *conv*: Calculate cumulus convection and convective rainfall.

[3] Amdahl's Law splits the code into sequential (F_s flops) and parallel (F_p flops) portions, where $F = F_s + F_p$, and states that the best achievable efficiency is $E = \frac{F}{N_p F_s + F_p} \times 100\,\%$. If the parallel portion of the code is not perfectly parallel (*e.g.* conditional code in GCM physics), the efficiency will be lower than this.

Table 1. Efficiency (% use) of physics routines under assumptions of MIMD or SIMD execution.

Routine	Serial flops	MIMD		SIMD		SIMD + Indexing	
		Flops	Efficiency	Flops	Efficiency	Flops	Efficiency
	F	M	E_M	S	E_S	S'	E'_S
radin	1,144,490	486	65	490	65	490	65
surfset	49,287	22	62	47	29	47	29
qsgam	2,386944	666	100	666	100	666	100
sflux	1,625226	719	63	997	45	997	45
radfs	155,646,207	53,899	80	64,039	68	63,571	68
surfupa	122,181	86	39	136	25	136	25
vertmix	4,292,303	3,476	34	3,583	33	3,515	34
gwdrag	1,892,464	530	99	530	99	530	99
cvmix	229,376	64	100	64	100	64	100
rainda	285,982	188	42	255	31	220	36
conv	300,537	385	21	648	13	429	20
surfupb	56,670	58	27	120	13	120	13

Table 2. Overall efficiencies (% use) for timesteps with and without the radiation calculation.

Efficiency for timestep	MIMD physics	MIMD routines	SIMD+ Indexing	SIMD
with radiation	78	67	66	65
without radiation	56	51	47	45

- *cvmix*: Calculate vertical mixing of momentum due to convection.
- *surfupb*: Update soil moisture and snow cover due to precipitation.

Table 1 contains a summary of the results obtained for the physics routines under the assumptions of SIMD (with or without indexing) or MIMD execution. SIMD execution with indexing refers to the index option available on some SIMD machines, where simultaneous execution is permitted for the same instructions applied to different data values[4]. Note that the recoding necessary to make use of this feature may be significant.

The only routines which benefited significantly from the use of the indexing option were the routines which calculate large scale and convective rain, *rainda* and *conv*. This is consistent with the fact that in these routines there are calculations which are performed identically at different gridpoints, but at different vertical levels. Overall, use of the indexing option made only a slight difference to the results obtained.

Comparing the MIMD and SIMD results in Table 1, we see that the largest differences between the MIMD and SIMD results occur for the surface updating routines, *surfset, surfupa* and *surfupb*, while a fairly large difference also occurs for *sflux*, which calculates the surface fluxes. This results from the fact that these routines contain code for updating land, sea and ice points separately—under MIMD execution, one PE may be updating the temperature at a land point while another is melting sea-ice, but in the SIMD model each PE is constrained by the single instruction requirement. However, none of these routines is particularly expensive, so their effect on the overall efficiency is relatively small.

Clearly the results in Table 1, which are based on one timestep which included a call to the radiation scheme, are dominated by the flops in the radiation scheme. In line with the usual practice of calling the radiation scheme every two to three hours of model time, we have in Table 2 presented overall efficiencies for timesteps which include or exclude the radiation calculation. The *MIMD physics* column refers to the best case where synchronization is assumed to occur only at the end of the entire physics calculation, while the *MIMD routines* column assumes synchronization after each routine. Table 2 shows that the overall performance penalty associated with a change from the best possible MIMD to SIMD was found to be only 15 % to 20 %, depending on whether the radiation calculation is performed and whether SIMD indexing is permitted.

The results for both SIMD and MIMD were improved by the relatively high efficiency of the radiation scheme which dominates the results at lower resolutions, and by the relatively small amount of work performed in inefficient routines such as *conv* and *surfupb*. Note that as model resolution increases, the dominance of the radiation scheme may be expected to decrease, as an increase in the model

[4] An example of this situation would be a segment of code in the cumulus convection routine which calculates moistening of the environment. This calculation might be performed at one level at one gridpoint and at a different level at another gridpoint. The index option allows simultaneous execution of the code at both gridpoints.

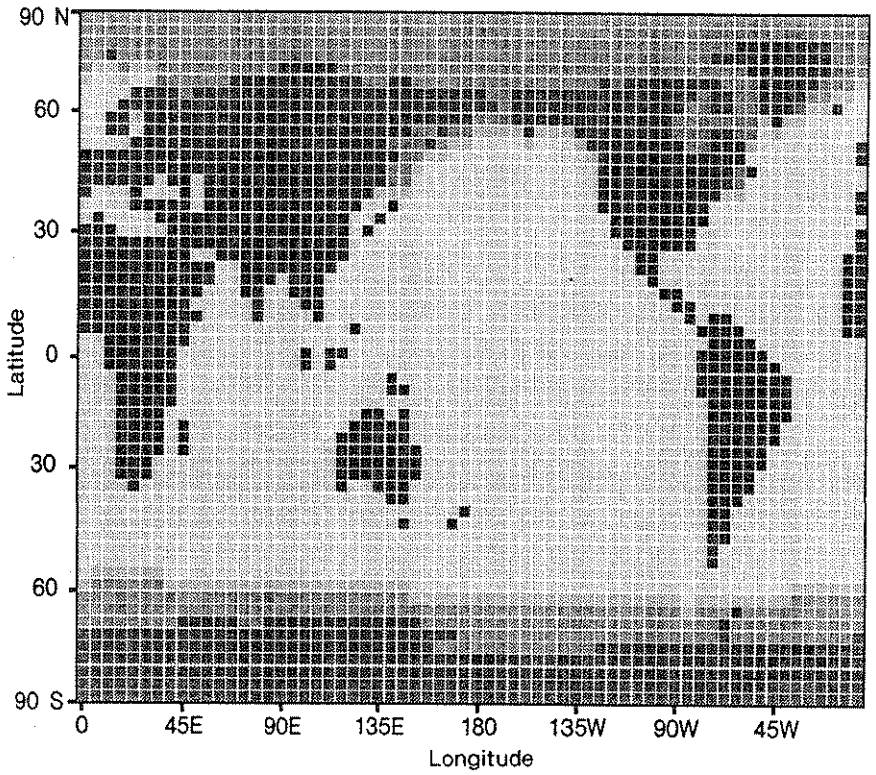

Fig. 1. Global map of flops at each gridpoint for the three surface updating routines (*surfset, surfupa, surfupb*) combined. Darker shading indicates more flops.

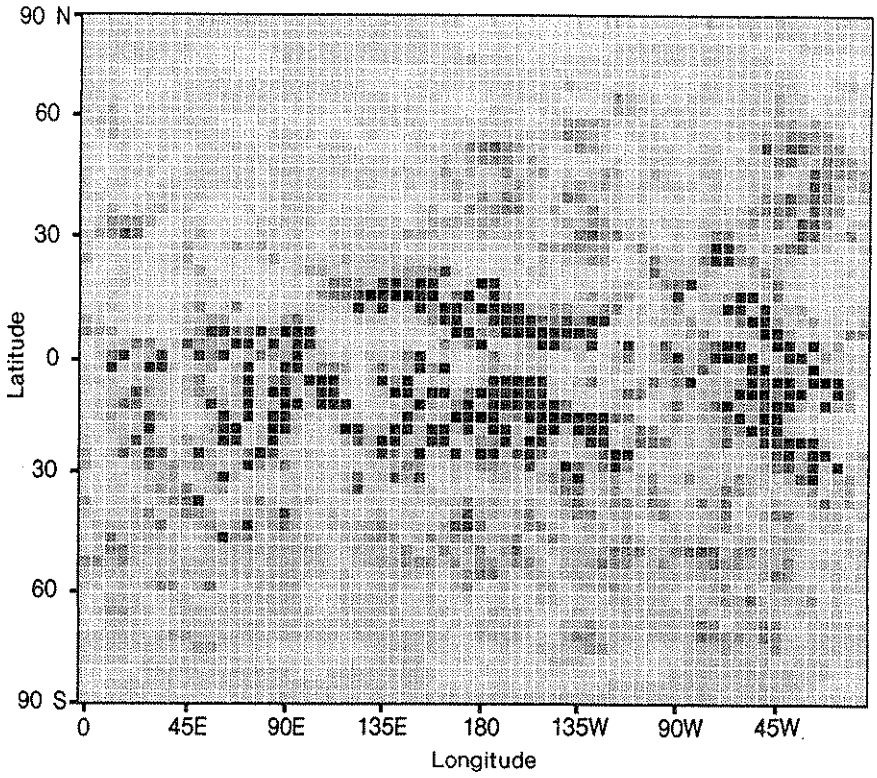

Fig 2. Global map of flops at each gridpoint for the convection routine (*conv*). Darker shading indicates more flops.

timestep causes the number of timesteps between calls to the radiation scheme to also increase. It was noted that the radiation scheme would have scored even better if it were not for the fact that the efficiency of the short-wave calculation was limited to a maximum of 50 %, because the sun only shines on about 50 % of the globe at any one time. In Figs. 1 and 2 we show maps of the global distribution of flops at each gridpoint for the three surface updating routines (Fig. 1) and for the cumulus convection routine *conv* (Fig. 2). Figure 1 shows clearly the difference between land and sea points, while in Fig. 2 the heavier computational load at low latitudes is readily apparent.

The physical parameterizations represent about 60 % of the flops in this version of CSIRO9, with most of the remainder being accounted for by the spectral (Legendre and fast Fourier) transforms and the dynamics. In contrast to the physics, the problems regarding parallelization of the dynamics and transforms relate mainly to communication. Whilst a number of authors (*e.g.* Foster *et al.*, 1992; Sato, 1992) have recently investigated the performance of the spectral transform method on massively parallel computers, it remains unclear what type of spatial (and temporal) discretization will ultimately perform best on massively parallel computers, whether MIMD or SIMD. However, our results regarding the SIMD efficiency of the physics should be applicable regardless of which type of discretization is chosen, as the physics is always done on the grid.

4. Conclusion

The SIMD and MIMD analyses described in this paper provide a good, architecture independent efficiency estimate of the physics routines of a GCM. The results obtained indicate that the physics in the CSIRO9 GCM can be scheduled with an acceptable level of efficiency on a SIMD architecture, if each PE is allocated to a single vertical column of the model. In particular, the performance penalty relative to the best possible MIMD execution with the same mapping of processors to gridpoints is a modest 15 % to 20 %, depending on whether the radiation calculation is performed and whether SIMD indexing is permitted. This penalty is quite small when one compares the cost/performance ratio of a SIMD machine such as the Maspar with that of comparable MIMD machines. As the physical parameterizations in CSIRO9 are comparable to those in other climate and NWP models, we expect that the physics in other models (whether spectral or finite difference, global or limited area) would also achieve acceptable performance. This is an encouraging result which suggests that SIMD climate modelling is feasible.

Acknowledgements

The authors thank John McGregor and Jack Katzfey of CSIRO Division of Atmospheric Research for their helpful comments on this paper.

This work was conducted as part of the Division of Information Technology High Performance Computation Program. The High Performance Computation Program is a joint program between the CSIRO Division of Information Technology and The Royal Melbourne Institute of Technology.

References

Chern, I.-L. and I. Foster, 1992: Alternative numerical methods and their parallel implementation, in *Computer Hardware Advanced Mathematics and Model Physics Pilot Project Final Report*, publication DOE/ER-0541T, 11–18.

Foster, I., W. Gropp and R. Stevens, 1992: The parallel scalability of the spectral transform method. *Mon. Wea. Rev.*, **120**, 835–850.

Francis, R., L. Rotstayn, D. Abramson and M. Dix, 1992: SIMD climate modelling, *Parallel Computing '91*, Evans, D., G. Joubert and H. Liddell Eds., Elsevier, 471–482.

Gordon, H. and B. Hunt, 1991: Droughts, floods and sea surface temperature anomalies: a modelling approach. *Int. J. Climatology*, **11**, 347–365.

Hunt, B. and H. Gordon, 1989: Diurnally varying regional climate simulations. *Int. J. Climatology*, **9**, 331–356.

Kauranne, T., 1990: An introduction to parallel processing in meteorology, *The Dawn of Massively Parallel Processing in Meteorology*, Hoffmann, G. and D. Maretis Eds., Springer Verlag, 3–20.

Sato, R., 1992: Implementation of a parallel community climate model SIMD Connection Machine 2 version, in *Computer Hardware Advanced Mathematics and Model Physics Pilot Project Final Report*, publication DOE/ER-0541T, 33–39.

Swarztrauber, P. and R. Sato, 1990: Solving the shallow water equations on the Cray X-MP/48 and the Connection Machine 2, in *The Dawn of Massively Parallel Processing in Meteorology*, Hoffmann, G. and D. Maretis Eds., Springer Verlag, 260–276.

Tuccillo, J., 1990: Numerical solution of the primitive equations on the Connection Machine, in *The Dawn of Massively Parallel Processing in Meteorology*, Hoffmann, G. and D. Maretis Eds., Springer Verlag, 340–361.

U.S. Department of Energy (DOE), 1990: *Building an Advanced Climate Model: Program Plan for the CHAMMP Climate Modeling Program*, publication DOE/ER-0479T, 70 pp.

SIMD 並列計算機での GCM 物理過程の実行適合性について

Leon Rotstayn
(CSIRO 大気研究部)
Rhys Francis • David Abramson
(CSIRO 情報技術部)
Martin Dix
(CSIRO 大気研究部)

次世代の大循環モデル (GCM) に必要とされるテラフロップス機 (毎秒 10^{12}回以上の浮動小数点演算ができる計算機) の一つの可能性は SIMD(Single Instruction stream, Multiple Data stream) 法による超並列計算機である。SIMD 法でモデルを走らせるうえでの問題は、モデルの物理過程のパラメタリゼーションがこの種の機械に対して効率良く計算できるようになっているかどうかである。並列機の各プロセッサがモデルの各グリッドに割り当てられると仮定して、SIMD 法で効率的に計算できるかどうかの可能性を評価するために、GCM の物理過程を解析した。この論文ではさまざまな物理過程ルーチンに対する結果を、MIMD(Multiple Instruction stream, Multiple Data stream) 法で計算を行なった場合に得られるものと比較した。その結果、SIMD 法の MIMD 法に対する非効率性は 15%~20%に過ぎないことが分かった。これは SIMD 法にとって満足の行く結果であり、気候モデルや数値予報にとって SIMD 計算機をしりぞけるべきでないことを意味している。