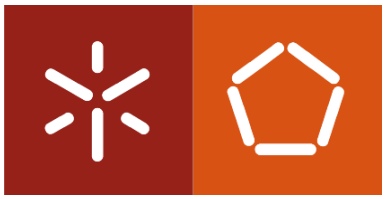
NAS parallel benchmarks

Hugo Gião
PG41073

April 21, 2020



Contents

1 Introdução

Este trabalho visa a explorar o desempenho de diversos *benchmarks* do pacote *NAS parallel benchmarks*, em vários sistemas. Foram utilizadas as versões sequenciais, *OpenMP*, *MPI* e híbrida desses *benchmarks* nos testes realizados.

Para realizar as medições do desempenho dos *benchmarks* foram utilizadas várias ferramentas de monitorização que permitiram monitorizar o estado do sistema durante a execução dos programas. Os resultados desses utilitários foram posteriormente tratados utilizando *python* e foram criados gráficos para o auxílio da análise dos resultados utilizando a biblioteca *matplotlib*.

Numa primeira fase foram realizados testes com o objetivo de analisar a performance dos *benchmarks* utilizando diferentes compiladores, versões e opções de otimização. Para tal escolhi alguns dos *benchmarks* disponíveis de modo a poder simular diversos perfis de execução e realizei as medições para os diferentes compiladores, versões e opções de otimização.

Numa segunda fase utilizei alguns dos *benchmarks* para medir o impacto no sistema do aumento da carga das aplicações. Para isso foram escolhidos alguns *benchmarks* e utilizadas várias ferramentas de monitorização de modo a observar o estado do sistema na execução de diferentes classes dos mesmos.

Este relatório está organizado primeiramente por uma secção de exposição da metodologia e *benchmarks* e testes utilizados, das diferentes opções, classes e *benchmarks* utilizados para cada uma das versões *NPB*, posteriormente contém uma secção dos resultados obtidos. Será enviado também um ficheiro contendo os resultados dos vários testes realizados.

2 Maquinas utilizadas

Os testes foram realizados em sistemas com diferentes características, esses pertencem a um dos seguintes *racks* do *SeARCH*:

- nós **r641:**
  - **CPU:** 2 X *E5-2650v2* cada um com 8 cores e *SMT*
  - **RAM:**64GiB
  - **L3 CACHE:** 2 x 20480KiB
  - **L2 CACHE:** 16 X 256KiB
  - **L1 CACHE:** 2 X 16 X 32KiB
- nós **r662:**
  - **CPU:** 2 x *E5-2695v2* cada um com 12 cores e *SMT*
  - **RAM:**64GiB
  - **L3 CACHE:** 2 x 30720KiB
  - **L2 CACHE:** 16 X 256KiB
  - **L1 CACHE:** 2 X 16 X 32KiB

3 Metodologia utilizada e testes realizados

Para cada um dos *benchmarks* corridos foram utilizadas um conjunto de métricas estas foram obtidas correndo cada *benchmark* um número de vezes dependendo do seu tamanho e guardar os resultados de diferentes comandos em ficheiros de modo a poder visualizar o estado do sistema durante a execução do programa. As medições foram repetidas 5 vezes e a média das 3 melhores execuções foi utilizada para criar os gráficos utilizados para a sua interpretação. Estes testes foram corridos utilizando o sistema *PBS* e para facilitar o uso de diferentes compiladores, opções e os uso de vários *benchmarks* foi utilizado *python* e biblioteca *os* para poder automatizar as invocações dos *scripts PBS* com diferentes parâmetros. Os ficheiros utilizados para obter os vários *benchmarks* são depois guardados com um nome e numa diretoria apropriada para depois serem utilizados para gerar os gráficos.

3.1 *Kernels* e pseudo-aplicações utilizadas

Utilizei os seguintes *kernels* nos testes realizados:

- **IS:** Utilizei este *kernel* para a realização dos testes Sequenciais, *OpenMp* e *MPI*.Este *kernel* permite avaliar acessos á memória.
- **EP:** Utilizado nos testes *OpenMp* e *Mpi*.Permite visualizar a performance duma aplicação com elevada escalabilidade.
- **MG:** Utilizado nos testes Sequenciais, *OpenMp* e *MPI*.Permite avaliar comunicação e performance em aplicações com uso intensivo de memória.
- **CG:** Utilizado nos testes Sequenciais. Permite medir acessos a memória irregulares.

Utilizei as seguintes pseudo-aplicações nos testes realizados:

- **BT:** Utilizado nos testes híbridos.
- **SP:** Utilizado nos testes híbridos.

3.2 Testes realizados

As metricas relativas aos tempos de execução,*system time*,*user time*,*idle*,*iowait* e utilização máxima de *CPU* foram obtidos utilizando o comando *sar* 1 essas são:

- **Tempo total** O tempo total de uma execução do programa em segundos foi obtido dividindo o numero de linhas contendo informação relativa ao uso de *CPU* do comando pelo numero de execuções do programa.
- **Tempo system** Esta métrica foi obtida somando os valores de *system time* em cada linha do comando multiplicados por 0.01 dividido pelo numero de execuções do programa.
- **Tempo user** Esta métrica foi obtida somando os valores de *user time* em cada linha do comando multiplicados por 0.01 dividido pelo numero de execuções do programa.
- **Tempo iowait** Esta métrica foi obtida somando os valores de *iowait* em cada linha do comando multiplicados por 0.01 dividido pelo numero de execuções do programa.
- **Tempo idle** Esta métrica foi obtida somando os valores de *idle* em cada linha do comando multiplicados por 0.01 dividido pelo numero de execuções do programa.
- **Utilização máxima de CPU** Foi obtida obtendo subtraindo 1 ao valor mínimo do campo de idle das diferentes linhas produzidas pelo comando.

As métricas relativas ao uso de memoria no sistema durante a execução do programa foram obtidas utilizando o comando $sar\ -r\ 1$.

- **Memoria utilizada KiB** Esta métrica foi obtida medindo o valor máximo do campo *memory used KB* produzido pelo comando.
- **Memoria utilizada percentagem** Esta métrica foi obtida medindo o valor máximo do campo *memory used per* produzido pelo comando.

Para as métricas relacionadas com o uso de rede e disco foi utilizado o comando $sar\ -b\ 1$.

- **Utilização de disco**Foi obtida somando os valores do campos $breads$ e $bwrtns$ produzidos pelo comando.
- **Utilização da rede** Foi obtida somando os valores do campo *tps* produzidos pelo comando.

Apesar de apenas ser possível verificar o estado de uma maquina utilizando os comandos acima,os testes foram realizados da mesma maneira para as varias versões.Os resultados obtidos apesar de não darem uma visão completa do sistema para as versões *MPI* e híbrida permitem observando uma da duas maquinas obter grande parte da informação que seria retirada se fossem realizadas medições em ambas as maquinas.

3.3 Testes,métricas,opções e compiladores utilizados para as diferentes versões dos testes *NPB*

3.3.1 Versão sequencial

As tabelas seguintes contém os vários *benchmarks*,classes, compiladores, níveis de otimização, versões dos compiladores e métricas utilizadas. Apenas utilizei uma versão da suite de compilação da Intel por ser a única das disponíveis no *SeARCH* com licença. Nos gráficos os compiladores, versões e nível de otimização utilizados são identificados da forma (compilador).(nível de otimização).(versão).

Suites de compilação	**Versões**	**Opções de optimização**
Intel	19.0.5.281(2019)	O1,O2,O3
GNU	5.3.0,6.1.0,7.2.0	O1,O2,O3

Benchmarks	**Tamanhos**	**Metricas**
IS,MG,CG	S,W,A,B,C	Tempo total Tempo system Tempo user Tempo iowait Tempo idle Utilização maxima de CPU Memória utilizada KiB Memoria utilizada percentagem Utilização de disco

Para a realização dos testes relativos a performance dos *Benchmarks* foram utilizados todos os compiladores, opções de otimização e versões acima. Todos os testes foram corridos com opção $mcmodel = medium$. Nestes testes foram também utilizadas todas as métricas acima exceto a utilização máxima de *CPU*.Foram utilizados os tamanhos B e C

Para os testes do impacto no sistema foram utilizados os compiladores de C e Fortran da GNU, versão 5.3.0 e opção -O3. Foram utilizadas as métricas Tempo total, Utilização máxima de *CPU*,Memoria utilizada *KiB*,Memoria utilizada percentagem e utilização de disco. Foram utilizadas as todas as classes na tabela.

Os testes relacionados com as versões sequenciais dos *benchmarks* foram corridos em apenas 1 maquina.

3.3.2 Versão OpenMP

Para estes *benchmarks* foram utilizados os mesmos compiladores e opções que anteriormente. Nos gráficos os compiladores versões e nível de otimização utilizados são identificados da forma (compilador).(nível de otimização).(versão).

Suites de compilação	**Versões**	**Opções de optimização**
Intel	19.0.5.281(2019)	O1,O2,O3
GNU	5.3.0,6.1.0,7.2.0	O1,O2,O3

Benchmarks	**Tamanhos**	**Métricas**
IS,MG,EP	S,W,A,B,C,D	Tempo total Tempo system Tempo user Tempo iowait Tempo idle Utilização máxima de CPU Memória utilizada KiB Memoria utilizada percentagem Utilização de disco

Para a realização dos testes relativos a performance dos *Benchmarks* foram utilizados todos os compiladores, opções de otimização e versões. Todos os testes foram corridos com opção *mcmodel = medium*. Nestes testes foram também utilizadas todas as métricas acima exceto a utilização máxima de *CPU*.

Para os testes do impacto no sistema foram utilizados os compiladores de C e Fortran da GNU, versão 5.3.0 e opção -O3. Foram utilizadas as métricas Tempo total, Utilização máxima de *CPU*, Memoria utilizada *KiB*, memoria utilizada percentagem e utilização de disco.

Os testes relacionados com as versões *OpenMp* dos *benchmarks* foram corridos em apenas 1 maquina.

3.3.3 Versão MPI

Para a realização dos testes dos *benchmarks MPI* foram utilizados varias versões das implementações *OpenMpi_ eth* e *Mpich_ eth* da *Intel* e da *GNU*.Os compiladores são identificados nos gráficos da forma (compilador).(versão).Nos gráficos o termo *gnu_ eth* refere-se á implementação *OpenMpi* da *GNU*,*intel_ eth* á implementação *OpenMpi* da *Intel* e os termos com *mpich* ou *mpich2* referem-se a essas implementações do respetivo fabricante.

Implementação de MPI	**Versões**	**Fabricantes**
OpenMpi_eth	1.8.2,1.6.3,	Intel,GNU
MPICH_eth	1.2.7,1.5	Intel,GNU

Benchmarks	**Tamanhos**	**Metricas**
IS,MG,EP	S,W,A,B,C,D	Tempo total Tempo system Tempo user Tempo iowait Tempo idle Utilização maxima de CPU Memória utilizada KiB Memoria utilizada percentagem Utilização de disco Utilização de rede

Para a realização dos testes relativos a performance dos *Benchmarks* foram utilizados todos os compiladores, opções de otimização e versões. Todos os testes foram corridos com opção médium. Nestes testes foram também utilizadas todas as métricas acima exceto a utilização máxima de *CPU*. Foram utilizadas as classes C e D.

Para os testes do impacto no sistema foram utilizados os compiladores de C e Fortran da GNU, versão 5.3.0 e opção -O3. Foram utilizadas as métricas Tempo total, Utilização máxima de *CPU*, Memoria utilizada *KiB*, memoria utilizada percentagem e utilização de disco. Foram utilizadas todas as classes na tabela.

Os testes *MPI* foram corridos utilizando os comandos $mpirun\ -np\ cores\ -mca\ btl\ self, sm, tcp\ bin$ para as versões *openmpi* da GNU e $mpirun\ -np\ cores\ bin$ para as restantes.

Os testes relacionados com as versões *MPI* dos *benchmarks* foram corridos em apenas 2 maquinas do mesmo rack.

3.3.4 Versão híbrida

Para os testes da versão híbrida foram utilizados varias versões das implementações *OpenMpi_ eth* da *Intel* e *GNU*, utilizando as implementações *MPICH* utilizadas para os testes *MPI* ocorriam problemas em termos de acesso as bibliotecas de *OpenMp*.Os compiladores são identificados nos gráficos da forma (compilador).(versão).Nos gráficos o termo *gnu_ eth* refere-se á implementação *OpenMpi* da *GNU*,*intel_ eth* á implementação *OpenMpi* da *Intel*

Implementação de MPI	**Versões**	**Fabricantes**
OpenMpi_eth	1.8.2(Intel e GNU) ,1.6.3(Intel) ,1.8.4(GNU),	Intel,GNU

Benchmarks	**Tamanhos**	**Metricas**
SP,BT	S,W,A,B,C	Tempo total Tempo system Tempo user Tempo iowait Tempo idle Utilização maxima de CPU Memória utilizada KiB Memoria utilizada percentagem Utilização de disco Utilização de rede

Para a realização dos testes relativos a performance dos *Benchmarks* foram utilizados todos os compiladores, opções de otimização e versões. Todos os testes foram corridos com opção *mcmodel=medium*. Nestes testes foram também utilizadas todas as métricas acima exceto a utilização máxima de *CPU*. Foram nestes testes utilizadas as classes A e B.

Para os testes do impacto no sistema foram utilizados os compiladores de C e *Fortran* da *GNU*, versão 5.3.0 e opção -O3. Foram utilizadas as métricas Tempo total, Utilização máxima de *CPU*, memoria utilizada *KiB*, memoria utilizada percentagem e utilização de disco. Foram nestes testes utilizadas todas as classes na tabela.

Os testes da versão híbrida foram corridos utilizando os comandos *mpirun −np cores −mca btl* $self, sm, tcp$ *bin* para as versões *openmpi* da GNU e *mpirun −np cores bin* para as restantes e antes de correr cada teste é invocado este comando *export* $OMP\_NUM\_THREADS = THREADS$ com $THREADS = 32$ para maquinas $r641$ e $THREADS = 48$ para maquinas $r662$

Estes testes foram corridos em duas maquinas do mesmo *rack*.

4 Analise dos resultados

4.1 Performance dos *benchmarks*

4.1.1 Versão sequencial

Os testes das versões sequenciais dos *benchmarks* mostram que é gasto um tempo considerável em *idle* isto acontece porque não são utilizados a maioria dos recursos de *CPU*, o restante tempo é gasto consideravelmente mais em *user time*, isto é visto também pela maior correlação entre as proporções dos diferentes valores nos testes de tempo total e *user time*, isto indica que os possíveis problemas da performance ocorrem devido a limitações dos *benchmarks*.Estes resultados foram observados para os vários *benchmarks* e tamanhos. O tempo de *iowait* é também consistentemente de baixa proporção comparativamente ao tempo total dos *benchmarks*.

Contrariamente ao que seria esperado utilizar opções de otimização mais elevados não conduz necessariamente a melhores resultados nos *benchmarks* .

4.1.2 Versão OpenMP

Nos testes *OMP* notei que para os testes da classe C que o compilador da Intel apresenta tempos substancialmente maiores para os testes IS e MG, nos testes EP pode-se verificar o oposto. Notei também que nos testes OMP existe uma maior correlação entre o tempo gasto em *user time* e tempo total em relação ao *system time*. Notei que os tempos em *idle* são proporcionalmente bastante inferiores do que os da versão sequencial. Em termos de uso de memoria não foram encontradas grandes disparidades entre as diferentes otimizações e compiladores.

Para os testes da classe D os compiladores da Intel continuam a ter tempos superiores nos testes IS e MG apesar de terem uma menor discrepância comparativamente ao tamanho inferior, para o teste EP os tempos foram consideravelmente menores para os compiladores da Intel tal como na classe menor em proporções similares. O uso de memoria continua similar para as diferentes opções de compilação e compiladores.

4.1.3 Versão Mpi

No *benchmark* IS da classe C podemos ver que os tempos de execução apesar de variarem com as implementações de *MPI* utilizadas também variam com as máquinas utilizadas. Podemos observar tempos superiores nas máquinas r641, nessas máquinas consegui observar uma tendência para melhores tempos nas implementações da Intel. Notei que os tempos de *iowait* são insignificantes e as distribuições dos tempos de *idle* e *user time* sãos similares as do tempo total. Sendo que o tempo gasto em *idle* é de proporção significativa. Em termos de escrita no disco encontrei que as implementações da Intel são mais eficientes. Em termos de memoria utilizada verifiquei que esta é bastante superior nas versões da Intel.

Para os testes do *benchmark* MG de classe C os tempos de execução são inferiores para as versões da Intel. Notei que os tempos de *user time* são insignificantes quando comparados com os de *user time* exceto para a versão *OpenMpi* de GNU versão 1.6.3.O tempo gasto em *idle* é também significativo sendo maior para as versões da GNU especialmente a versão *OpenMpi* 1.6.3. Verificamos em ambas as máquinas que o número de escritas em disco é superior para as versões *OpenMpi* de *GNU* comparativamente as outras. Notei semelhantemente ao *benchamrk* anterior que o uso de memoria é superior nas implementações da *intel*.

Para o *benchmark* EP da classe C, notei maiores tempos de execução para as máquinas r641 nas implementações da GNU, nas máquinas r662 não encontrei essas diferenças. Notei que foi gasto mais tempo em *user time* relativamente ao system time sendo que o tempo gasto em *system time* é insignificante com a exceção mais uma vez do compilador *OpenMpi* versão 1.6.3 da GNU.O número de blocos escritos é superior nas implementações *OpenMpi* da *GNU*. Verifiquei que as implementações da *Intel* utilizam mais memória.

No *benchmark* IS da classe D podemos verificar que os tempos de execução são similares para todos as implementações. Os tempos de *usertime* seguem uma distribuição similar ao tempo total. Podemos notar mais uma vez que o tempo de *system time* da implementação *OpenMpi* 1.6.3 da GNU é bastante superior aos de todas as outras implementações neste caso notei também que as versões da *Intel* tém maiores *sytem time* do que as restantes versões da GNU. Notei em termos de tempo *idle* tempos muito superiores para as máquinas r662 relativamente as restantes, nestas máquinas esses tempos são semelhantes nas máquinas r641 notei tempos inferiores para as versões da Intel. Mais uma vez em termos de leituras de disco o as implementações de *OpenMpi* da GNU são superiores. Os usos de memoria são similares em todas as versões.

No *benchmark* MG da classe D notei tempos ligeiramente melhores para as implementações da Intel. Mais uma vez *system time* muito maior do que as outras implementações para a implementação *OpenMpi* da GNU versão 1.6.3. Em termos de *iowait* os valores não são significativos apesar disso os valores de *iowait* observados são em ambas as máquinas consideravelmente superiores para as implementações *OpenMpi* da GNU. Similarmente a outros testes notei tempos em *idle* bastante superiores nas máquinas r662 relativamente as r641, em ambas notei também tempos *idle* bastante inferiores para as implementações da *intel*. O numero de escritas no disco é superiores para as implementações *OpenMpi* da GNU.O uso de memoria é semelhante.

Os resultados do *benchmark* EP são semelhantes aos do MG para a classe D, com a exceção do uso de memória que é superior para as implementações da *Intel*.

Em suma podemos concluir que para os vários tamanhos e *benchmarks* consegui obter geralmente melhores resultados utilizando as implementações de *OpenMpi* e *MPICH* da Intel. Sendo que essas demonstram realizar um melhor uso dos recursos.

4.1.4 Versão híbrida

No *benchmark* SP da classe A, podemos notar uma anomalia em que o tempo de execução utilizando a implementação *OpenMpi* da *GNU* versão 1.6.3 nas maquinas r662,sendo que os tempos são bastante superiores as restantes implementações e aos da mesma implementação nas maquina r641.Com essa exceção notei tempos menores nas implementações da Intel para ambos os benchmarks. Notei também um menor número de escritas e leituras nos compiladores da *Intel* e maior uso de memoria nos mesmos. Notei também maiores tempos de *system time* e *user time* para os compiladores da *Intel* e menores de *iowait* e *idle* sendo que os de *idle* representam uma taxa significativa do tempo global.

4.2 Impacto no sistema

4.2.1 Versão sequencial

Em termos de tempo podemos notar que o crescimento do mesmo é mais acentuado para os *benchmarks* IS e MG. Notei também que para os testes IS e MG o uso de memória é crescente quando o tamanho dos *benchmarks* é aumentado. Para o *benchmark* CG o uso de memória varia pouco com o tamanho dos *benchmarks*. Em termos de blocos escritos no disco os valores crescem com os testes para o *benchmark* CG, nos outros *benchmarks* podemos notar repetidamente um maior número de escritas para os testes da classe A, B, C em relação aos outros.

4.2.2 Versão OpenMP

Em termos de tempo da aplicação podemos notar um aumento do tempo de execução elevado para todos os testes com o aumento de carga. Em termos de utilização máxima de *CPU* notei que esta é maior do que as do A e B nas classes S e W, mas estas são inferiores do que as dos testes C e D para o *benchmark* IS. No *benchmark* MG temos uma utilização máxima de *CPU* com pouca variação. No *benchmark* EP a utilização de*CPU* aumenta até ao tamanho B onde estagna. Em termos de uso de memória o uso de memória é significativamente maior para o tamanho D e valores similares para os outros tamanhos no *benchmark* IS. Algo similar acontece para o *benchmark* MG.No *benchmark* EP o uso de memória tem pouca variação independentemente das classes dos *benchmarks*. Em termos de utilização de disco encontrei que esta foi maior para a classe C para o *benchmark* IS. O mesmo foi encontrado para o *benchmark* MG e EP.

4.2.3 Versão Mpi

Para o teste IS encontrei um crescimento acentuado dos tempos de execução com o tamanho de carga. Em termos de uso máximo de *CPU* notei que este se situa perto dos 100% nas máquinas r641 em todos os *benchmarks* e possui valores oscilatórios nas máquinas r662 sendo maior nas duas classes maiores, mas não chegando a 70%. O uso de memoria apesar de ter alguma oscilação aumenta com o aumento da carga. O numero de blocos lidos são superiores nas classes S e W diminuem passando para a classe A e depois aumentam até a classe D. Notei também que o número de *packets* enviados é superior nas classes S e W nas maquinas r662 e depois inferior nas classes A,B e C com a classe D tendo um valor de *packets* superior aos das ultimas nas maquinas r641 e sendo o máximo nas maquinas r662.

Nos testes MG notei mais uma vez uso de *CPU* perto dos 100% para as máquinas r641 e com valores oscilatórios com tendências a ter maiores valores com o aumento de carga, mas sempre menos de 70% mas maquinas r662.A utilização de memoria aumenta com a carga. Em termos de blocos escritos este é superior para a classe D e maior nas classes S e W do que nas restantes em ambas as máquinas. O número de *paquets* enviados e recebidos é superior nas classes S e W nas máquinas r641 e na classe D nas máquinas r662.

No testes EP notei usos de *CPU* crescentes com a carga estagnando a partir da classe B, apesar disso estes estão mais uma vez perto dos 100% nas maquinas r641 e não chegam aso 70% nas maquinas r662.Em termos de memória encontrei sos maiores nas classes A,B e C para as maquinas r641 e usos similares para as maquinas r662.Em termos de leituras de disco encontrei valores bastante superiores para a classe D nas maquinas, seguidas das classes S e W. Em termos de transmissões *tcp* nas máquinas

r641 as classes S e W tem valores maiores seguidos da classe D e depois as restantes, nas máquinas r662 a classe D tem valores maiores do que as classes S e W.

4.2.4 Versão híbrida

Notei uma utilização de *CPU* muito baixa isto poderá ser devido a uma má configuração. Para o teste SP podemos notar um incremento do uso dos vários recursos com o aumento da carga este sendo mais acentuado no uso de disco e rede. Em termos de memória e *CPU* o uso tende a estagnar por volta dos 4-5% dependendo da máquina para o *CPU* e 3% para memória nas duas máquinas. Para o teste BT obtive resultados semelhantes, salientando que os testes da classe S utilizam ainda mais recursos de rede e disco proporcionalmente as restantes classes que no teste anterior.