

Parallel sorting by regular sampling

SRIR

1. Opis algorytmu

Algorytm „Równoległego sortowania przez regularne dzielenie” został wymyślony przez Li.

Oryginalnie algorytm składa się z czterech faz:

1. Faza I

Sortowanie swojego zestawu danych przez każdy proces

2. Faza II

Jeden z procesów zbiera od innych procesów i siebie wartości równe liczbie procesów, które są wybierane w sposób regularny w każdym z procesów. Kolejno sortuje te elementy i wybiera „liczba procesów - 1” elementów, które przesyłane są do kolejnych procesów. Następnie każdy proces partycjonuje swój zbiór danych na odpowiednią liczbę podzbiorów w oparciu o otrzymane *pivoty* od serwera.

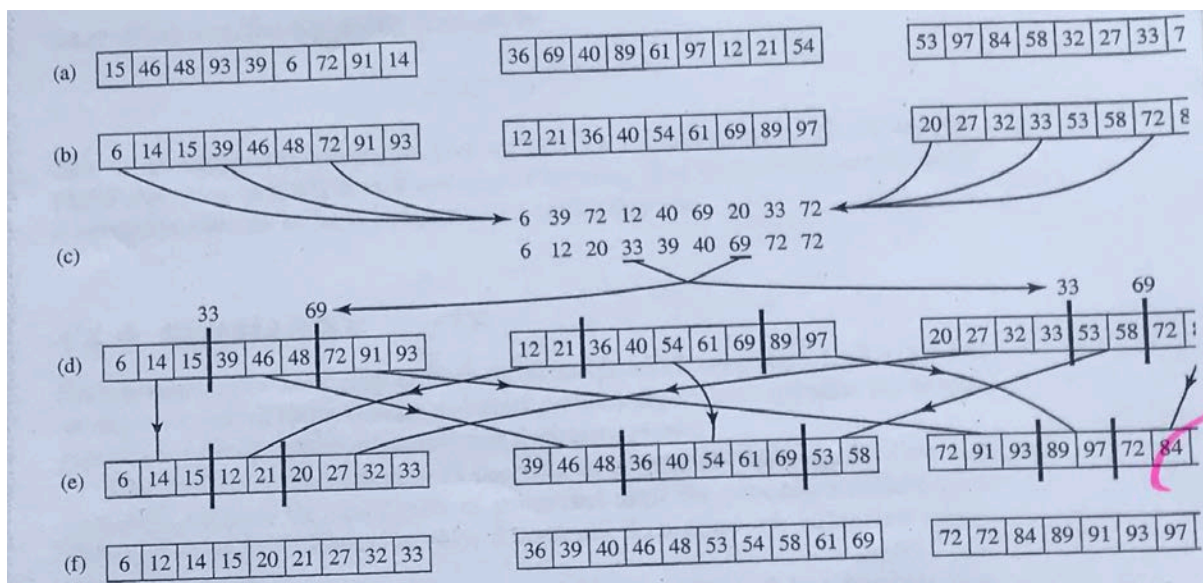
3. Faza III

W tej fazie, każdy z procesów zachowuje „swoją” (przypisaną do danego procesu) porcję danych i przesyła pozostałe dane do reszty procesów.

4. Faza IV

Każdy z procesów łączy zebrane dane z każdego procesu w jedną listę i je sortuje (*multimerge*).

W fazie tej maksymalna liczba elementów, które dany proces musi złączyć wynosi $2 * \text{liczba elementów} / \text{liczba procesów}$



Rys. 1 Diagram przedstawiający przebieg algorytmu

Wydajność algorytmu:

p - liczba procesów

n - liczba elementów

Dla obliczeń

$$O\left(\frac{n}{p} * \log\left(\frac{n}{p}\right) + p^2 * \log\left(\frac{n}{p}\right) * \log(p)\right)$$

$$\text{Dla } n \gg p: O\left(\frac{n}{p} * (\log(n) + \log(p))\right)$$

Dla komunikacji:

$$O(\log(p) + n/p)$$

$$\text{Dla } n \gg p: O(n/p)$$

Funkcja skalowalności:

$$p^C / p = p^{(C-1)}$$

Algorytm ten posiada trzy zalety względem algorytmu

hyperquicksort:

- Przechowuje listę rozmiarów w bardziej zbalansowany sposób pomiędzy procesami
- Pomija niepotrzebną komunikację pomiędzy kluczami
- Nie wymaga, aby liczba procesów była potęgą liczby 2

Ponadto algorytm umożliwia wprowadzenie prostej komunikacji *all-to-all*, która może być zaimplementowana w taki sposób, aby każdy element był przenoszony tylko raz.

2. Opis struktury projektu

Projekt składa się z 3 plików:

- utilities.h + utilities.c
- PSRC.c

W plikach utilities znajdują się metody wykorzystywane w algorytmie do wypisywania macierzy, porównywania oraz łączenia tablic (multimerge).

Plik PSRC.c to właściwy kod realizujący algorytm, w komentarzach w kodzie mamy wydzielone fazy, które są opisane w punkcie 3.

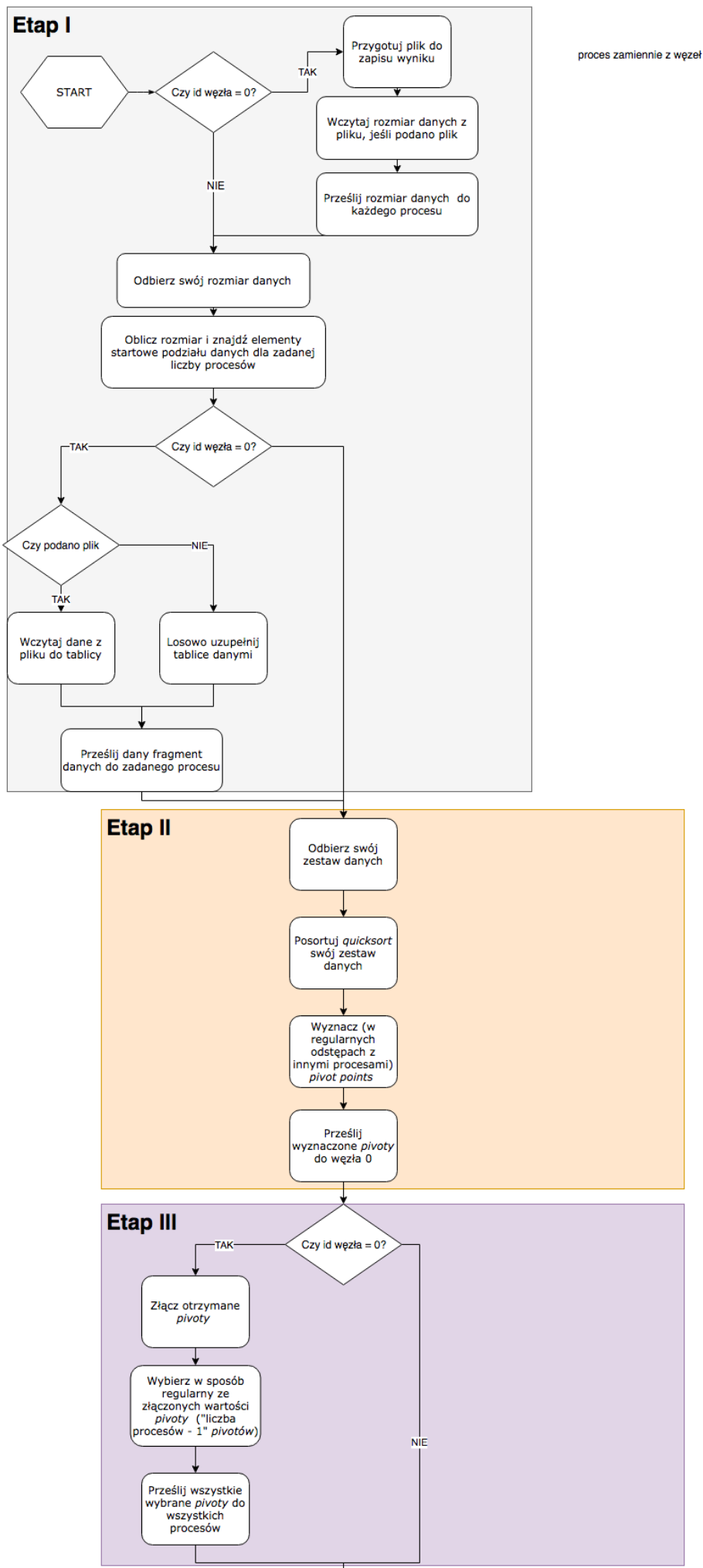
Ponadto w projekcie umieszczono katalog z przykładowymi danymi: *sample*, gdzie input to dane wejściowe do posortowania, a output wynik oczekiwany

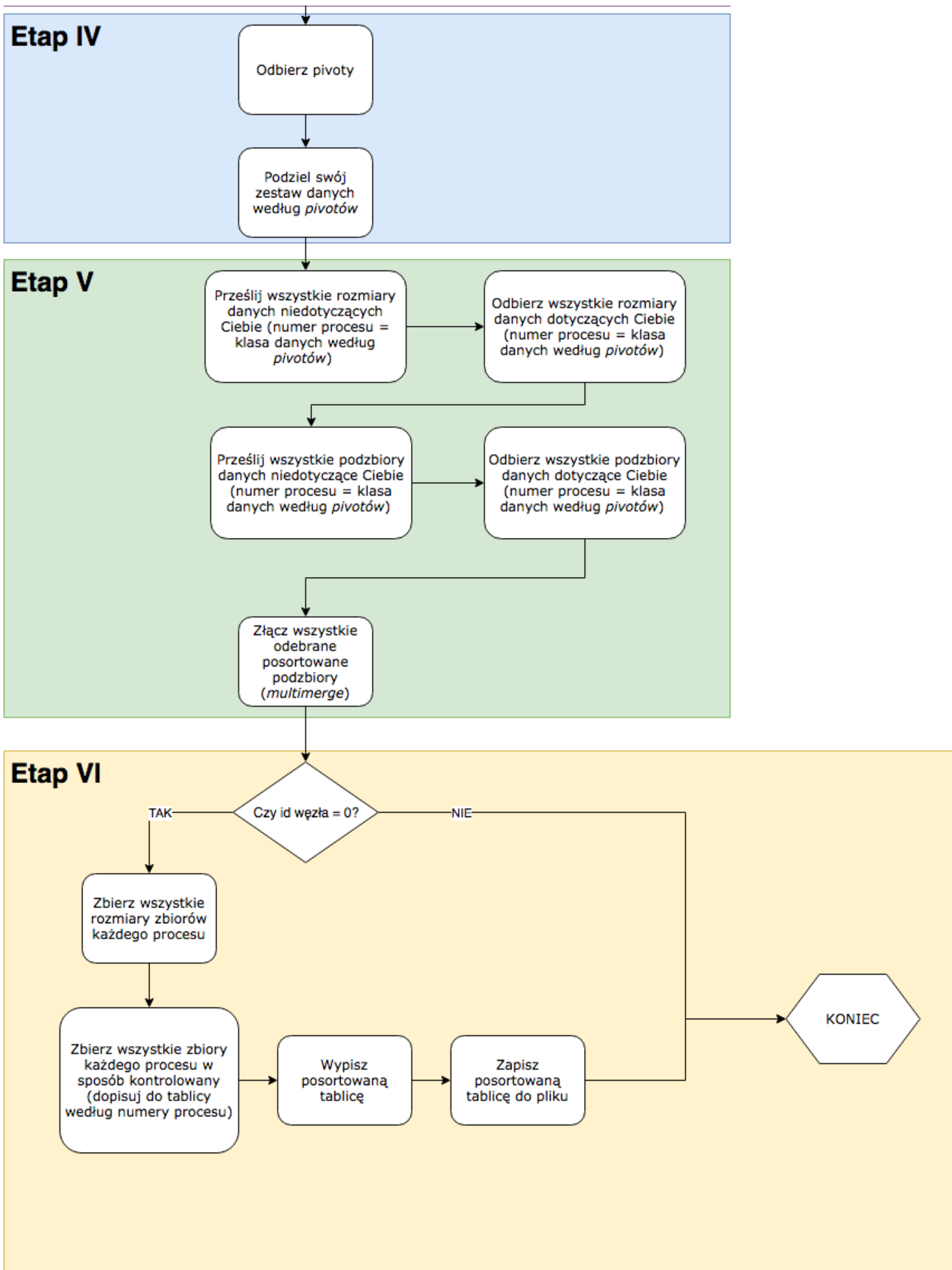
W projekcie znajduje się również katalog doc z dokumentacją.

Ponadto do projektu dołączono makefile, którego metody opisane zostały w punkcie 4

3. Opis działania

Opis działania przedstawiony został na diagramie Diag. 1, Diag. 2
Diagram został również umieszczony w folderze doc/





Aplikacja składa się z 6 etapów, które pokrywają się z etapami algorytmu przedstawionymi w punkcie 1.

1. ETAP1 - Początkowo aplikacja przygotowuje dane do posortowania
2. ETAP 2 - W kolejnej fazie, dane są rozdzielane na procesy i wybierane są wartości, będące *pivotami*
3. ETAP 3 - Następnie pivoty są przesłane do serwera, który je łączy i wybiera „liczba procesów - 1” pivotów. Pivoty te są przesyłane do każdego procesu.
4. ETAP 4 - Kolejno pivoty te są wykorzystywane przez każdy proces do podziału swoich danych na partycje
5. ETAP 5 - W następnym kroku każdy proces pozyskuje tylko dane z zadanej przez identyfikator partycji od każdego procesu i je łączy.
6. ETAP 6 - Na końcu dane te są przesyłane do serwera, który je łączy i wypisuje do pliku

4. Opis obsługi programu

Kompilacja programu i jego uruchomienie odbywa się przy wykorzystaniu makefile'a.

Udostępnia on 4 reguły:

- *make all* zbudowanie projektu
- *make clean* przywrócenie folderu do stanu początkowego (usunięcie plików .o oraz .out)
- *make rebuild* wyczyszczenie i zbudowanie projektu
- *make run N=<liczba procesów> FILE==<ścieżka do pliku> NODES=<nodes>*

- uruchomienie aplikacji podając odpowiednio parametry: N oraz opcjonalnie FILE i NODES, gdzie

N - liczba procesów (domyślnie 4),

FILE to nazwa pliku do wczytania do procesowania,

NODES to wskazanie na plik z adresami węzłów

np: *make run N=4 FILE==sample/input_50.data NODES=nodes*

- uruchomi program na 4 procesach i na tylu węzłach ile w pliku nodes, wczytując dane z pliku o ścieżce sample/input_50.data

Kroki do skompilowania i uruchomienia na jednym węźle:

1. Przejdź do katalogu z projektem i plikiem make: MPI_PSRS
2. Wykonaj *make clean*
3. Wykonaj *make run N=<liczba procesów> FILE=<ścieżka do pliku>*

4. Przykład wyjścia z programu

Na rys. 2 oraz rys. 3 przedstawiono wyjście z programu dla danych z pliku.

```
tlaz@tlaz:~/Projects/MPI_PSRS$ mpxec -n 4 ./PSRS_xmp.out -f sample/input_50.data

Starting
Number of processes 4

Size of whole data to process: 50
[0] Table values to sort:
7 16 3 84 20 17 47 40 3 69 92 96 74 32 53 43 81 73 31 13 91 88 21 98 54 52 36 67 49 78 95 84 75 46 88 41 76 2 83 4 8 77 49 93 40 80 97 18 26 36
The end
[process-0] myDataLength=12, myDataStarts=0
[process-2] myDataLength=12, myDataStarts=24
[process-0] myData 3 3 7 16 17 20 40 47 69 84 92 96
[process-0] pivots 3 16 40 84
[process-1] myDataLength=12, myDataStarts=12
[process-3] myDataLength=14, myDataStarts=36
[process-2] myData 36 41 46 49 52 54 67 75 78 84 88 95
[process-2] pivots 36 49 67 84
[process-1] myData 13 21 31 32 43 53 73 74 81 88 91 98
[process-1] pivots 13 32 73 88
[process-3] myData 2 4 8 18 26 36 40 49 76 77 80 83 93 97
[process-3] pivots 2 18 49 80
[process-0] server pivots 3 16 40 84 13 32 73 88 36 49 67 84 2 18 49 80
[process-0] Merged pivots 2 3 13 16 18 32 36 40 49 49 67 73 80 84 84 88
[process-0] Chosen pivots 18 49 80
[process-3] Chosen pivots on node 18 49 80
[process-3][class-0] class start=0, class length=4
[process-1] Chosen pivots on node 18 49 80
[process-1][class-0] class start=0, class length=1
[process-1][class-1] class start=1, class length=4
[process-1][class-2] class start=5, class length=3
[process-1][class-3] class start=8, class length=4
[process-2] Chosen pivots on node 18 49 80
[process-2][class-0] class start=0, class length=0
[process-2][class-1] class start=0, class length=4
[process-2][class-2] class start=4, class length=5
[process-2][class-3] class start=9, class length=3
[process-0] Chosen pivots on node 18 49 80
[process-0][class-0] class start=0, class length=5
[process-0][class-1] class start=5, class length=3
[process-0][class-2] class start=8, class length=1
[process-0][class-3] class start=9, class length=3
[process-3][class-1] class start=4, class length=4
[process-3][class-2] class start=8, class length=3
[process-3][class-3] class start=11, class length=3
[0] tmp: 0 5 | new: 0 5
[2] tmp: 8 1 | new: 0 1
[1] tmp: 5 3 | new: 0 3
[0] tmp: 0 1 | new: 5 1
[2] tmp: 5 3 | new: 1 3
[2] tmp: 4 5 | new: 4 5
[1] tmp: 1 4 | new: 3 4
[1] tmp: 0 4 | new: 7 4
[2] tmp: 8 3 | new: 9 3
[1] tmp: 4 4 | new: 11 4
[3] tmp: 9 3 | new: 0 3
[0] tmp: 0 0 | new: 6 0
[3] tmp: 8 4 | new: 3 4
[0] tmp: 0 4 | new: 6 4
[3] tmp: 9 3 | new: 7 3
[3] tmp: 11 3 | new: 10 3
[process-2] Data for myid partition 52 53 54 67 69 73 74 75 76 77 78 80
[process-1] Data for myid partition 20 21 26 31 32 36 36 40 40 41 43 46 47 49 49
[process-3] Data for myid partition 81 83 84 84 88 88 91 92 93 95 96 97 98
[process-0] Data for myid partition 2 3 3 4 7 8 13 16 17 18
[process-0] sorted data 2 3 3 4 7 8 13 16 17 18 20 21 26 31 32 36 36 40 40 41 43 46 47 49 49 52 53 54 67 69 73 74 75 76 77 78 80 81 83 84 84 88 88 91 92 93 95 96 97 98

Clock time (seconds) = 0.001031

The end
tlaz@tlaz:~/Projects/MPI_PSRS$
```

Rys. 2 Przykładowe wyjście z programu dla
N=4 FILE=sample/input_50.data

```
tiastlcat -r/Projects/WP1_P855 nqexec -n 2 -fPSM5_omp_out -f sample/Output_500.data

Starting
Number of processes 2

Size of whole data to process: 500
[0] Table values to sort:
370 210 680 523 157 880 807 47 383 926 379 695 684 936 460 207 909 217 890 694 379 829 207 734 114 936 304 658 399 423 409 484 85 691 154 564 770 236 364 448 248 683 214 276 248 956 284 381 468 504 574 71 842 17 463 723 473 625 708 482 451 119 724 154 793 244 687 746 173 821 719 873 364 37 483 716 358 627
451 880 262 167 183 230 501 949 478 863 399 752 788 238 941 338 784 231 754 31 281 653 63 388 883 837 67 422 221 487 686 988 188 412 472 219 530 235 648 136 280 167 567 121 764 883 568 52 271 334 653 886 3 238 366 888 341 882 748 747 164 65 93 136 835 388 861 648 693 77 487 394 339 147 158 713 568 995 64
772 993 87 182 71 83 919 631 325 671 146 907 436 655 247 301 697 123 989 66 27 474 691 963 876 710 996 199 648 987 926 97 715 924 509 289 219 523 78 884 48 343 241 5 761 836 568 568 877 933 651 283 698 828 568 886 597 978 471 76 358 387 193 887 455 96 941 783 786 368 686 822 652 88 540 838 154 623 974 61
8 638 583 297 482 654 388 688 941 785 714 848 977 455 782 182 652 796 97 575 96 968 461 883 627 6 951 990 289 343 791 938 495 563 575 668 902 491 783 995 589 165 368 922 555 693 136 423 858 389 442 768 511 657 928 448 786 113 455 862 563 132 534 645 443 741 338 248 417 893 771 681 864 613 378 682 253 581
227 448 415 389 223 411 439 481 715 645 518 778 788 986 188 228 202 6 2 230 179 48 887 711 781 483 238 234 535 587 28 459 493 477 241 788 621 453 37 418 418 853 538 388 818 204 114 423 787 13 832 528 718 65 277 711 624 383 387 514 48 523 84 283 44 299 188 82 31 822 12 788 648 881 239 883 528 88 993 261 81
5 918 217 203 16 662 511 111 67 179 547 71 11 867 760 881 980 581 452 146 197 557 441 377 873 318 988 508 915 133 643 23 380 597 648 888 772 389 152 869 272 234 682 76 463 480 916 73 375 695 941 223 448 763 879 53 887 758 321 562 589 858 737 648 883 119 85 774 862 848 372 739 152 744 886 792 89 666 886 87
387 597 228 143 351 592 755 339 547 143 788 818 656 986 646 782 596 534 455 99 386 979 241 446 575 354
The end
[process-0] myDataLength=258, myDataStart=8
[process-1] myDataLength=258, myDataStart=258
[process-0] myData 3 5 17 27 31 37 48 47 48 52 43 44 45 46 47 71 76 77 78 83 84 85 93 96 97 187 188 188 116 119 121 133 136 146 147 154 156 154 157 158 164 164 164 167 173 181 182 193 199 287 287 218 214 219 219 223 238 231 235 238 239 248 241 244 244 247 242 271 276 288 288 281 283 284 296 297 381 384
387 398 317 325 329 334 338 341 343 358 358 388 378 379 379 383 383 389 398 394 396 399 482 493 487 489 510 422 432 438 431 431 433 433 463 468 468 471 472 473 474 478 482 484 388 381 389 389 318 323 323 339 348 388 388 388 378 374 383 397 686 886 886 886 886 818 818 812 812 823 823 821 821 837 838 848 9
48 651 652 653 654 655 656 664 671 683 684 685 688 691 691 691 694 695 697 688 785 786 788 710 711 714 715 716 719 721 724 734 748 746 747 752 754 761 764 772 776 788 783 784 793 883 883 887 887 888 821 822 828 829 835 836 837 838 848 848 842 861 866 866 869 873 876 877 888 882 883 888 887 985 916 919 924 924
926 933 936 941 941 949 958 957 987 974 977 978 981 987 989 991 995 996
[process-1] myData 2 6 12 12 12 12 16 18 21 31 37 48 48 44 53 65 67 71 75 76 88 82 84 87 87 89 93 96 97 99 111 113 114 119 132 133 143 143 143 148 152 152 165 179 179 182 183 188 200 202 203 203 206 217 223 228 231 234 241 241 248 250 253 272 277 281 282 286 289 291 291 294 297 299 303 309 316 318 328 330 331 343
358 351 354 368 372 375 377 378 387 388 389 397 401 411 411 415 417 423 441 442 443 443 446 448 449 452 453 455 463 483 488 491 493 495 495 508 586 587 588 511 511 523 526 531 534 534 539 541 547 547 555 555 557 558 562 563 563 575 575 581 581 588 592 596 597 597 613 619 628 621 621 622 627 648 643 6
82 685 686 686 686 687 687 688 688 687 687 688 688 693 695 700 704 707 710 711 711 715 727 737 739 741 744 755 758 760 760 763 763 770 771 772 774 782 788 789 792 796 796 798 887 888 888 810 810 818 818 815 821 822 828 828 832 833 836 837 838 848 842 848 858 858 861 862 862 864 866 867 869 869 873 873 876 877 879 888 881 882 883 883 883 886 887 896 891 893 897 902 907 909 910 915 916 916 918 918 919 922 922 924 924 926 926 928
928 928 938 941 951 953 956 968 964 975 978 986 978 988 993 988 995 995
[process-0] pivots 3 584
[process-1] pivots 2 534
[process-0] server pivots 3 384 2 534
[process-0] Merged pivots 2 3 584 534
[process-0] Chosen pivots 584
[process-0] Chosen pivots on node 504
[process-1] Chosen pivots on node 584
[process-1][[class-0] class start=0, class length=117
[process-1][class-1] class start=117, class length=133
[process-0][[class-0] class start=0, class length=126
[process-0][class-1] class start=126, class length=124
[0] tmp: 0 126 1
[0] tmp: 0 126 1 new: 0 126
[process-0] Data for myid partition 2 3 5 6 4 12 12 13 16 17 27 28 31 31 37 37 48 48 44 47 48 52 53 63 64 65 65 66 67 67 71 71 71 73 76 76 77 78 80 82 83 84 84 85 85 87 89 93 93 96 96 97 97 99 187 188 188 111 113 114 114 119 119 121 132 133 133 136 143 143 146 147 148 152 152 154 154 157 158 164 164 164 16
4 164 165 167 172 179 181 182 183 188 193 199 200 202 203 203 206 207 218 214 217 219 219 223 223 228 228 231 231 234 235 238 239 240 241 241 241 244 244 247 248 250 253 262 271 272 276 277 280 280 281 281 282 283 284 286 289 291 291 294 296 297 297 299 301 303 304 307 388 389 316 317 318 325 328 329 330
328 329 338 331 334 338 341 343 343 358 358 351 354 358 368 368 372 375 377 378 378 379 381 383 387 388 388 389 390 394 396 397 399 401 402 403 407 408 411 411 415 416 417 422 423 433 441 442 443 443 446 448 448 449 451 451 452 453 455 455 455 455 461 463 468 468 471 472 473 474 478 482 483 484 498 491 493 495 49
3 493 495 495 500 500 501 504
[process-0] sorted data 2 3 5 6 4 12 12 13 16 17 27 28 31 31 37 37 48 48 44 44 47 48 52 53 63 64 65 65 66 67 67 71 71 71 73 76 76 77 78 80 82 83 84 84 85 85 87 89 93 93 96 96 97 97 99 187 188 188 111 113 114 114 119 119 121 132 133 133 136 143 143 146 147 148 152 152 154 154 157 158 164 164 164 16
5 167 172 179 179 181 182 182 183 188 193 199 200 202 203 203 206 207 218 214 217 219 219 223 223 228 228 231 231 234 235 238 239 240 241 241 241 244 244 247 248 250 253 262 271 272 276 277 280 280 281 281 282 283 284 286 289 291 291 294 296 297 297 299 301 303 304 307 388 389 316 317 318 325 328 329 330
331 334 338 341 343 343 358 358 351 354 358 368 368 372 375 377 378 378 379 381 383 387 388 388 389 390 394 396 397 399 401 402 403 407 408 411 411 415 416 417 422 423 433 441 442 443 443 446 448 448 449 451 451 452 453 455 455 455 455 461 463 468 468 471 472 473 474 478 482 483 484 498 491 493 495 49
3 500 500 501 504 504 507 509 509 511 511 516 523 523 528 533 534 534 539 539 540 541 547 547 555 555 557 558 568 568 562 563 563 567 568 568 568 578 574 575 575 581 581 583 588 592 596 597 597 597 604 606 606 607 608 610 613 613 619 628 621 621 622 622 623 625 627 631 637 638 648 643 645 645 646 646 648
648 649 651 652 653 654 655 656 657 658 659 668 662 664 666 668 671 677 681 682 682 683 684 685 688 691 691 691 693 694 695 695 697 698 700 704 705 706 707 708 710 710 711 711 711 714 715 715 716 716 719 721 724 727 734 737 738 748 741 744 746 747 752 754 755 758 760 760 761 761 763 764 770 771 772 772 774 77
6 788 782 783 784 788 789 792 793 796 798 883 887 887 887 888 888 810 810 815 821 822 828 828 832 833 836 837 838 848 842 848 858 858 861 862 862 864 866 867 869 869 873 873 876 877 879 888 881 882 883 883 883 886 887 896 891 893 897 902 907 909 910 915 916 916 918 918 919 922 922 924 924 926 926 928
928 928 938 941 942 942 949 951 953 956 956 957 960 964 967 974 975 977 978 979 981 986 987 988 990 991 991 995 996 998 999 999
[1] tmp: 126 124 1 new: 0 124
[1] tmp: 127 123 1 new: 124 123
[process-1] Data for myid partition 586 887 589 480 511 511 516 523 523 528 526 531 534 534 539 539 540 541 547 547 555 555 557 558 568 568 562 563 563 567 568 568 568 578 574 575 575 581 581 583 588 592 596 597 597 597 604 606 606 607 608 610 613 613 619 628 621 621 622 623 625 627 631 637 638 648 643 64
5 645 646 648 648 649 651 652 653 653 658 659 659 662 661 666 668 671 677 681 682 682 683 684 685 688 691 691 691 693 694 695 695 697 698 700 704 705 706 707 708 710 710 711 711 711 714 715 715 716 716 719 721 724 727 734 737 738 748 741 744 746 747 752 754 755 758 760 760 761 761 763 764 770 771 772 772 774 77
6 777 772 772 774 776 788 782 783 784 788 789 792 793 796 798 883 887 887 887 888 888 810 810 815 821 822 828 828 832 835 836 837 838 848 842 848 858 858 861 862 862 866 866 867 869 869 873 873 876 877 879 888 881 882 883 883 883 886 887 896 891 893 897 902 907 909 910 915 916 916 918 918 919 922 922 924 924 926 926 928
928 928 928 928 932 932 939 941 941 941 949 951 953 956 956 957 960 964 967 974 975 977 978 979 981 986 987 988 990 991 991 995 996 998 999 999
[1] tmp: 126 124 1 new: 0 124
[1] tmp: 127 123 1 new: 124 123
[process-1] Data for myid partition 586 887 589 480 511 511 516 523 523 528 526 531 534 534 539 539 540 541 547 547 555 555 557 558 568 568 562 563 563 567 568 568 568 578 574 575 575 581 581 583 588 592 596 597 597 597 604 606 606 607 608 610 613 613 619 628 621 621 622 623 625 627 631 637 638 648 643 64
5 645 646 648 648 649 651 652 653 653 658 659 659 662 661 666 668 671 677 681 682 682 683 684 685 688 691 691 691 693 694 695 695 697 698 700 704 705 706 707 708 710 710 711 711 711 714 715 715 716 716 719 721 724 727 734 737 738 748 741 744 746 747 752 754 755 758 760 760 761 761 763 764 770 771 772 772 774 77
6 777 772 772 774 776 788 782 783 784 788 789 792 793 796 798 883 887 887 887 888 888 810 810 815 821 822 828 828 832 835 836 837 838 848 842 848 858 858 861 862 862 866 866 867 869 869 873 873 876 877 879 888 881 882 883 883 883 886 887 896 891 893 897 902 907 909 910 915 916 916 918 918 919 922 922 924 924 926 926 928
928 928 928 928 932 932 939 941 941 941 949 951 953 956 956 957 960 964 967 974 975 977 978 979 981 986 987 988 990 991 991 995 996 998 999 999
Clock time (seconds) = 0.008517

The end
```

Rys. 3 Przykładowe wyjścia z programu dla
N=2 FILE=sample/input_500.data

Wyjście z programu prezentuje:

- liczbę procesów,
- rozmiar danych do sortowania
- tablice wejściową z danymi
- rozmiar danych do posortowania na danym procesie wraz z indeksem startowym
- posortowane dane na danym procesie
- pivoty na danym procesie
- pivoty wybrane globalnie
- podział danych na danym procesie ze względu na pivoty wraz z długością danych w danej partycji
- aktualne dane na procesie

- posortowane dane
- czas działania algorytmu

Na rysunkach rys. 4, rys. 5, rys. 6 oraz rys. 7 niektóre komentarze zostały wyłączone dla większej ilości danych w celu zwiększenia czytelności.

```
klazgblaz~/Projects/001_PSR5 rxptec -n 4 -fPSR5.out -f sample/input_500.data

Starting
Number of processes 4

Size of whole data to process: 500
(0) Table values to sort:
510 210 680 523 157 688 807 47 383 926 378 695 684 916 468 207 009 317 890 694 370 820 207 734 114 936 304 658 399 433 409 484 85 691 154 664 776 296 164 448 244 683 214 276 240 956 284 381 468 594 574 71 842 17 463 721 473 625 708 482 451 110 724 154 753 244 687 746 173 821 719 873 164 37 403 716 350 637 451 888 262 107 181 239 581 940 478 869 379 752 789 238 981
318 784 231 754 31 281 957 63 388 883 837 67 422 223 807 900 900 108 612 472 210 539 235 648 516 286 167 547 121 764 883 568 52 771 134 653 866 3 238 396 888 341 882 740 747 164 65 93 136 835 108 861 685 691 77 407 394 329 147 158 711 568 995 64 772 993 84 182 71 83 910 631 325 671 146 907 416 655 247 301 697 133 989 66 27 474 651 967 876 718 996 199 648 987 924 97
715 924 509 280 219 523 78 684 40 343 241 5 761 836 568 568 877 933 651 283 698 818 568 866 597 978 471 76 318 387 193 807 455 96 941 783 706 368 686 822 652 48 540 838 154 623 974 610 638 583 297 482 654 389 688 941 705 714 840 977 455 282 182 622 796 97 575 96 960 461 883 627 6 951 999 289 343 291 918 495 563 975 660 902 491 761 999 589 165 360 922 555 693 316 4
12 658 388 442 700 511 657 698 449 296 113 455 862 963 132 534 645 443 741 330 748 417 803 771 685 964 631 378 682 233 881 727 649 452 550 231 411 495 403 715 645 531 770 708 956 183 620 205 6 2 250 179 40 887 711 704 443 288 291 355 387 21 657 493 817 841 198 462 453 27 411 619 953 558 888 818 294 115 621 787 11 831 526 710 63 277 711 924 383 387 941 89 923 84 203
44 299 188 82 31 922 12 789 668 891 939 883 926 88 991 281 815 918 217 283 16 462 511 111 67 179 547 71 12 867 760 881 990 581 452 148 297 557 441 377 873 318 906 500 915 133 643 53 200 597 640 888 772 389 152 869 272 234 682 76 483 490 916 73 375 695 941 223 448 763 879 93 887 758 331 562 589 850 737 646 883 119 85 774 862 848 372 739 152 744 886 792 89 666 286 8
7 397 597 228 143 351 592 755 539 547 143 798 810 656 986 646 782 596 534 455 99 506 979 241 446 575 354
The end
[process-0] sorted data 2 3 5 6 6 12 12 13 16 17 27 28 31 31 37 37 40 40 44 47 48 52 53 63 64 65 66 67 67 71 71 71 73 76 76 77 78 80 82 83 84 84 85 85 87 89 93 93 96 96 97 97 99 107 108 108 111 113 114 114 119 119 121 121 132 133 133 136 143 143 146 147 148 152 152 154 154 154 157 158 164 164 164 165 167 173 173 179 179 181 182 182 183 188 193 199 200 202 203 203 206
507 807 210 214 217 219 219 223 223 228 230 231 231 234 235 238 239 240 241 241 241 244 244 247 248 250 253 262 271 272 276 277 288 288 281 281 282 283 284 286 289 291 291 294 296 297 297 299 301 301 304 304 307 308 309 316 317 318 325 328 329 330 331 334 338 341 343 343 350 350 351 354 358 360 364 372 375 377 378 378 379 379 381 383 387 388 389 389 389 394 396 397 39
9 401 402 403 407 409 411 411 415 416 417 422 423 433 441 442 443 443 446 446 448 449 451 451 452 453 455 455 455 455 461 463 468 468 471 472 473 474 478 482 483 484 490 491 493 495 495 500 500 501 504 506 507 509 509 511 511 516 523 523 523 526 531 534 534 539 539 540 541 547 547 555 555 557 558 560 560 562 563 563 567 568 568 568 570 574 575 575 581 581 583 589 5
91 596 597 597 597 604 606 606 607 608 610 622 613 619 620 621 622 622 623 625 627 631 637 638 640 643 645 645 646 646 648 648 649 651 652 653 654 655 656 657 658 659 660 662 664 666 668 671 677 681 682 682 683 684 685 688 691 691 691 693 694 695 695 697 698 700 704 705 706 707 708 710 710 711 711 714 715 715 716 719 721 724 727 734 737 739 740 741 744 746 747
752 754 755 758 760 760 761 761 763 764 770 771 772 772 774 776 780 782 783 784 788 789 792 793 796 798 803 807 807 807 808 808 810 815 821 822 822 828 829 832 835 836 837 838 840 842 848 850 858 861 862 862 866 866 867 867 869 869 873 873 876 877 879 880 881 882 883 883 883 883 886 887 890 891 893 897 902 907 909 918 915 916 916 918 919 922 922 924 924 926 926 928
933 936 939 941 941 941 949 951 953 956 956 957 960 964 967 974 975 977 978 979 981 986 987 989 990 991 991 995 996 998 999 999
The end
Clock time (seconds) = 0.000171
```

Rys. 4 Przykładowe wyjścia z programu dla

N=4 FILE=sample/input_500.data

```
klazgblaz~/Projects/001_PSR5 rxptec -n 8 -fPSR5.out -f sample/input_500.data

Starting
Number of processes 8

Size of whole data to process: 500
(0) Table values to sort:
510 210 680 523 157 688 807 47 383 926 378 695 684 916 468 207 009 317 890 694 370 820 207 734 114 936 304 658 399 433 409 484 85 691 154 664 776 296 164 448 244 683 214 276 240 956 284 381 468 594 574 71 842 17 463 721 473 625 708 482 451 110 724 154 753 244 687 746 173 821 719 873 164 37 403 716 350 637 451 888 262 107 181 239 581 940 478 869 379 752 789 238 981
318 784 231 754 31 281 957 63 388 883 837 67 422 223 807 900 900 108 612 472 210 539 235 648 516 286 167 547 121 764 883 568 52 771 134 653 866 3 238 396 888 341 882 740 747 164 65 93 136 835 108 861 685 691 77 407 394 329 147 158 711 568 995 64 772 993 84 182 71 83 910 631 325 671 146 907 416 655 247 301 697 133 989 66 27 474 651 967 876 718 996 199 648 987 924 97
715 924 509 280 219 523 78 684 40 343 241 5 761 836 568 568 877 933 651 283 698 818 568 866 597 978 471 76 318 387 193 807 455 96 941 783 706 368 686 822 652 48 540 838 154 623 974 610 638 583 297 482 654 389 688 941 705 714 840 977 455 282 182 622 796 97 575 96 960 461 883 627 6 951 999 289 343 291 918 495 563 975 660 902 491 761 999 589 165 360 922 555 693 316 4
12 658 388 442 700 511 657 698 449 296 113 455 862 963 132 534 645 443 741 330 748 417 803 771 685 964 631 378 682 233 881 727 649 452 550 231 411 495 403 715 645 531 770 708 956 183 620 205 6 2 250 179 40 887 711 704 443 288 291 355 387 21 657 493 817 841 198 462 453 27 411 619 953 558 888 818 294 115 621 787 11 831 526 710 63 277 711 924 383 387 941 89 923 84 203
44 299 188 82 31 922 12 789 668 891 939 883 926 88 991 281 815 918 217 283 16 462 511 111 67 179 547 71 12 867 760 881 990 581 452 148 297 557 441 377 873 318 906 500 915 133 643 53 200 597 640 888 772 389 152 869 272 234 682 76 483 490 916 73 375 695 941 223 448 763 879 93 887 758 331 562 589 850 737 646 883 119 85 774 862 848 372 739 152 744 886 792 89 666 286 8
7 397 597 228 143 351 592 755 539 547 143 798 810 656 986 646 782 596 534 455 99 506 979 241 446 575 354
The end
[process-0] sorted data 2 3 5 6 6 12 12 13 16 17 27 28 31 31 37 37 40 40 44 47 48 52 53 63 64 65 66 67 67 71 71 71 73 76 76 77 78 80 82 83 84 84 85 85 87 89 93 93 96 96 97 97 99 107 108 108 111 113 114 114 119 119 121 121 132 133 133 136 143 143 146 147 148 152 152 154 154 154 157 158 164 164 164 165 167 173 173 179 179 181 182 183 188 193 199 200 202 203 203 206
507 807 210 214 217 219 219 223 223 228 230 231 231 234 235 238 239 240 241 241 241 244 244 247 248 250 253 262 271 272 276 277 288 288 281 281 282 283 284 286 289 291 291 294 296 297 297 299 301 301 304 304 307 308 309 316 317 318 325 328 329 330 331 334 338 341 343 343 350 350 351 354 358 360 364 372 375 377 378 378 379 379 381 383 387 388 389 389 389 394 396 397 39
9 401 402 403 407 409 411 411 415 416 417 422 423 433 441 442 443 443 446 446 448 449 451 451 452 453 455 455 455 455 461 463 468 468 471 472 473 474 478 482 483 484 490 491 493 495 495 500 500 501 504 506 507 509 509 511 511 516 523 523 523 526 531 534 534 539 539 540 541 547 547 555 555 557 558 560 560 562 563 563 567 568 568 568 570 574 575 575 581 581 583 589 5
91 596 597 597 597 604 606 606 607 608 610 622 613 619 620 621 622 622 623 625 627 631 637 638 640 643 645 645 646 646 648 648 649 651 652 653 654 655 656 657 658 659 660 662 664 666 668 671 677 681 682 682 683 684 685 688 691 691 691 693 694 695 695 697 698 700 704 705 706 707 708 710 710 711 711 714 715 715 716 719 721 724 727 734 737 739 740 741 744 746 747
752 754 755 758 760 760 761 761 763 764 770 771 772 772 774 776 780 782 783 784 788 789 792 793 796 798 803 807 807 807 808 808 810 815 821 822 822 828 829 832 835 836 837 838 840 842 848 850 858 861 862 862 866 866 867 867 869 869 873 873 876 877 879 880 881 882 883 883 883 883 886 887 890 891 893 897 902 907 909 918 915 916 916 918 919 922 922 924 924 926 926 928
933 936 939 941 941 941 949 951 953 956 956 957 960 964 967 974 975 977 978 979 981 986 987 989 990 991 991 995 996 998 999 999
The end
Clock time (seconds) = 0.259315
```

Rys. 5 Przykładowe wyjścia z programu dla

N=8 FILE=sample/input_500.data

The image displays a highly distorted and noisy document page. The text is rendered as a dense, illegible pattern of characters and symbols, likely due to severe corruption or a scanning artifact. The overall appearance is that of a corrupted or heavily distorted document page, with no discernible content or structure visible.

Na rysunkach rys. 4, rys. 5, rys. 6 oraz rys. 7 najważniejszy jest czas wykonania programu, ilość węzłów/procesów.

5. Źródła

Quinn, zad. 15.5, Grama, Gupta, rozdz. 13

<http://cswweb.cs.wfu.edu/bigiron/LittleFE-PSRS/build/html/PSRSalgorithm.html>

<http://cswweb.cs.wfu.edu/bigiron/LittleFE-PSRS/build/html/PSRSAppendixI.html>