Claims

1. Compiler system (1) for optimized compiling and auto-parallelization of a program code (3) for execution by a parallel processing system (2) having a plurality of processing units (21) simultaneously processing instructions on data in the parallel processing system (2) by executing the program code (3), wherein the compiler system (1) comprises means for translating a source program code written in a programming language into a machine code (3/32) as target programming language by generating the machine code (32) as parallel processing code comprising a number of instructions executable by the plurality of processing units (21) of the parallel processing system (2) or controlling the operation of the plurality of processing units (21);

wherein the execution of the processing code (32) by the parallel processing system (2) comprises the occurrence of latency times (26), the latency times being given by idle times of the processing units (21) between transmitting data back to the parallel processing system (2) after having processed a specific block of instructions of the processing code (32) on the data by the processing unit (21) and receiving data necessary for execution of a consecutive block of instructions of the processing code (32) by said processing units (21),

wherein the compiler system (1) comprises a parser module (11) for translating the source programming language (31) into a code (32) with a flow of elementary instructions executable by the processing units (21), the elementary instructions selectable out of a processing-unit-specific set of elementary instructions and the elementary instructions comprising elementary arithmetic operations (321) and/or logic operations (322) and/or control operations (325) and/or memory operations (325) for the number of processing units (21), characterized;

in that the parser module (11) comprises means for partitioning the code (32) of elementary instructions into computation block nodes (333), each computation block node (333) consisting of a smallest possible segmentation of a non-further decomposable sequence of elementary instructions of the code (32) processable by a single processing unit (21), the smallest possible segmentation of the elementary instructions being characterized by a sequence of elementary instructions framed by consecutive read and write instructions, said sequence being not further decomposable by smaller sequences of elementary instructions between consecutive

read and write instructions, and the read and write instructions needed to receive data required for processing said sequence of elementary instructions by the processing unit (21) and transmit back data after processing by the sequence;

in that the compiler system (1) comprises a matrix builder (15) for generating computation and transfer matrices (151,...,15i) out of computation chains (34) portioned from the code (32), wherein a computation chain (34) is formed by one or more computation block nodes (333) creating an ordered flow of computation block nodes (333), wherein each computation chain (34) is executed by one processing unit (21);

in that the computation matrix contains in each row the computation chain (34) of a processing unit (21) with each column having the sequence of elementary instructions of a computation block node (333) within the computation chain (34) of the row and each cell of the transfer matrix contains transfer properties associated with a data transfer from one to a consecutive computation block node (333), the transfer properties at least comprising what transfers are required at the beginning and at the end of each corresponding cell in the computation matrix; and

in that the compiler system (1) comprises an optimizer module (16) using an optimization technique by building different combinations of rows from the computation and the transfer matrices, each of the different combinations representing a possible machine code (32) as parallel processing code, and by minimizing an overall occurring latency time as aggregated latency times (26) providing an optimized combination as structure of computational chains (34) being processed by the plurality of processing units (21), wherein by means of a code generator (17) an optimized machine code is generated for the plurality of processing units of the parallel processing system having a minimized overall latency time (26).

2. Compiler system according to claim 1, characterized in that the source programming language is a high-level programming language.

3. Compiler system according to claim 2, characterized in that the high-level programming language comprises C and/or C++ and/or python and/or java.

4. Compiler system according to one of the claims 1 to 3, characterized in that the transition buffer units comprise cache memory modules and/or L1-cache.

5. Compiler system according to one of the claims 1 to 4, characterized in that the latency time comprises register access time and/or L1-cache access time and/or memory access time and/or I/O operation time and/or data network transfer time and/or processor configuration time.

6. Compiler system according to one of the claims 1 to 5, characterized in that at least two of the processing units have different sets of elementary instructions.

7. Compiler system according to one of the claims 1 to 6, characterized in that the arithmetic operations comprise the operations addition, subtraction, multiplication, and division.

8. Compiler system according to one of the claims 1 to 7, characterized in that the logic operations comprise a number of logical expressions like equal, not equal, greater than, less than, greater than or equal, less than or equal.

9. Compiler system according to one of the claims 1 to 8, characterized in that the control operations comprise a "branch expression" and/or a "loop expression".

10. Compiler system according to one of the claims 1 to 9, characterized in that the computation block nodes (333) consist of connected operation nodes (332), the operation nodes (332) being composed in one chain (34) having one or more input data nodes (331/3311) and one or more output data nodes (331/3312).

11. Compiler system according to claim 10, characterized in that the computation block nodes (333) are connected to next computation block nodes (333) or to control flow nodes (334) or to one or more input data nodes (331/3311) and one or more output data nodes (331/3312) building the chain (34).

12. Compiler system according to one of the claims 1 to 11, characterized in that each pair of computation and transfer matrices is realized forming a single numerical matrix and/or multi-dimensional numerical tensor, respectively.

13. Compiler system according to claim 12, characterized in that the numerical matrix optimization is realized as tensor optimization.

14. Compiler system according to one of the claims 12 or 13, characterized in that the numerical matrix optimization or tensor optimization by means of the optimizer module is based on a dedicated applied machine learning structure.

15. Compiler system according to one of the claims 1 to 14, characterized in that each computation block node (333) of a computation chain (34) is numbered along the ordered flow of computation block nodes (333) within the computation chain (34) by assigning block numbers to the computation block nodes (333), wherein in the computation and transfer matrix, the block numbers of the computation block nodes (333) are equal to the column of the matrices, and wherein the computation block nodes (333) with the same block number are distributed to different rows of the same column.

16. Compiler system according to one of the claims 1 to 15, characterized in that in the computation matrix, each row contains a computation chain (34) of a processing unit (21), wherein the cells of the computation matrix comprise the computation block nodes (333) of the computation chain (34), and wherein in the transfer matrix, the cells contain the required transfer properties to and/or from other computation block nodes (333).

17. Compiler system according to claim 16, characterized in that the transfer matrix with the cells having the transfer properties comprise at least of what information is needed in another computation block node (333).