Internal distribution code:

(A) [-] Publication in OJ
(B) [-] To Chairmen and Members
(C) [-] To Chairmen
(D) [X] No distribution

Datasheet for the decision of 2 April 2019

Case Number: T 1337/17 - 3.5.06

Application Number: 07015490.1

Publication Number: 1852777

IPC: G06F9/44

Language of the proceedings: EN

Title of invention:
Computer-aided parallelizing of computation graphs

Applicant:
Ab Initio Technology LLC

Headword:
Sort value indicator/AB INITIO

Relevant legal provisions:
EPC 1973 Art. 84

Keyword:
Claims - clarity (no)

Decisions cited:

Catchword:

Europäisches Patentamt
European Patent Office
Office européen des brevets

Beschwerdekammern

Boards of Appeal

Chambres de recours

Boards of Appeal of the
European Patent Office
Richard-Reitzner-Allee 8
85540 Haar
GERMANY
Tel. +49 (0)89 2399-0
Fax +49 (0)89 2399-4465

Case Number: T 1337/17 - 3.5.06

D E C I S I O N
of Technical Board of Appeal 3.5.06
of 2 April 2019

Appellant: (Applicant)	Ab Initio Technology LLC 201 Spring Street Lexington, MA 02421 (US)
Representative:	Lloyd, Patrick Alexander Desmond Reddie & Grose LLP The White Chapel Building 10 Whitechapel High Street London E1 8QS (GB)
Decision under appeal:	**Decision of the Examining Division of the European Patent Office posted on 10 January 2017 refusing European patent application No. 07015490.1 pursuant to Article 97(2) EPC.**

Composition of the Board:

Chairman	W. Sekretaruk
Members:	M. Müller
	S. Krischer

Summary of Facts and Submissions

I. The appeal is against the decision of the examining division dated 10 January 2017 to refuse European patent application No. 07 015 490 for lack of inventive step, Article 56 EPC 1973, as the mere computer implementation of a purely mathematical method.

II. A notice of appeal was filed on 10 March 2017, the appeal fee being paid on the same day. A statement of grounds of appeal was received on 19 May 2017. The appellant requested that the decision be set aside and a patent be granted on the basis of application documents on file, i.e. in particular on the basis of claims 1-13 dated 27 October 2016.

III. In an annex to a summons to oral proceedings, the board informed the appellant of its preliminary opinion that the claimed invention was not clear or sufficiently disclosed, Articles 83 and 84 EPC 1973. An inventive-step objection was also raised because the claimed subject-matter seemed not to show a technical effect, Article 56 EPC 1973.

IV. In response to the summons, the appellant filed, by letter dated 4 March 2019, sets of amended claims 1-8 according to a main and an auxiliary request and requested the grant of a patent on that basis.

Claim 1 of the main request reads as follows:

"A computer-implemented method for processing data in a computation graph, the method including, by a computer system:

passing work elements from each of multiple instances of a parallel first component (210) of the

computation graph to an inter-component link (205) of the computation graph, wherein the work elements are sorted according to a first key, in which the inter-component link includes a parallel partition element (221) configured to partition according to the first key and a parallel sorted merge element (231) configured to sort according to a second key;

receiving, by an instance (220) of the parallel partition element (221), a portion of the work elements from one of the parallel instances of the parallel first component (210), the received portion of the work elements being sorted according to the first key;

partitioning, by the instance (220) of the parallel partition element (221), the portion of the work elements among a plurality of flows according to the first key, wherein each flow of the plurality of flows is configured to receive a partition of the portion of the work elements having a first key of a particular value and is configured to flow to a respective instance (230) of the parallel sorted merge element (231), and

passing, by the parallel partition element (221), a sort value indicator related to the sort order according to the first key on each of the plurality of flows;

wherein the sort value indicator identifies a place in the sort order for the work elements such that subsequent work elements on the corresponding flow occur no earlier than the identified place in the sort order;

by an instance (230) of the parallel sorted merge element (231), in response to receiving the sort value indicator, merging the work elements received on each flow input to the parallel sorted merge element (231) and partitioned according to the first key, and sorting the merged work elements according to the second key to

generate merged outputs comprising a partition of the work elements according to the first key, including determining whether work elements received from each flow input can be passed to the merged outputs based on the sort value indicator of each flow input; and

passing the merged outputs comprising the partition of the work elements according to the first key to an instance (130) of a parallel second component (240) of the computation graph, downstream of the parallel first component (210), wherein the instance of the parallel second component (240) requires that its inputs be sorted according to the second key

wherein different instances of the parallel partition elements (221) and/or different instances of the parallel sorted merge elements (231) are configured to be executed by respective different coprocessors of the computing system."

Claim 1 of the auxiliary request is identical to that of the main request up to the passage stating what the "sort value indicator identifies" and, from thereon, reads as follows:

"... determining, by an instance of the parallel sorted merge element (231), in response to receiving a sort value indicator on a flow input of the plurality of flows, that pending work elements that precede the place in the sort order identified by the sort value indicator are received from each flow input;

by an instance (230) of the parallel sorted merge element (231), in response to receiving the sort value indicator, merging the work elements received on each flow input to the parallel sorted merge element (231) and partitioned according to the first key, and sorting the merged work elements according to the second key to generate merged outputs comprising a partition of the

work elements according to the first key, including determining whether work elements received from each flow input can be passed to the merged outputs based on the sort value indicator of each flow input; and

passing the merged outputs comprising the partition of the work elements according to the first key to an instance (130) of a parallel second component (240) of the computation graph, downstream of the parallel first component (210), wherein the instance of the parallel second component (240) requires that its inputs be sorted according to the second key, and

wherein different instances of the parallel partition elements (221) and/or different instances of the parallel sorted merge elements (231) are configured to be executed by respective different coprocessors of the computing system."

V. Oral proceedings took place as scheduled. At their end, the chairman announced the decision of the board.

Reasons for the Decision

The invention

1. The application generally relates to computation graphs in which the vertices and the links define, respectively, "data processing elements" and the data flow between them (see paragraphs 2 and 3).

1.1 The invention means to propose a general procedure for the transformation of "serial computation graphs" into equivalent "parallel" ones (see figure 3 and paragraph 53).

1.2 The data flow in parallel computation graphs is represented using data flow elements referred to as "partition", "gather" and "interconnection" (see figures 1B and 1C, nos. 115 and 135; figures 2B and 2C, no. 225). Several types of each of these elements are anticipated (see paragraphs 30-32, 35 and 49).

2. The invention in this divisional application is specifically concerned with a "sorted merge" component disclosed in paragraphs 70-75 of the divisional application as filed.

2.1 Figure 2C depicts an "interconnection network" linking m parallel instances of A with n instances of B via m parallel 1:n partitioning elements (nos. 220, 221) and n parallel m:1 gathering elements (nos. 230, 231). In total, the network has m*n inputs and n*m outputs and, likewise, in total, n*m serial links (effectively, each A with each B; see paragraphs 38 and 39).

2.2 The "sorted merge" element (see paragraphs 70-75), "assumes" that its inputs are sorted according to a particular sort order and that the output should remain sorted in that way (see paragraph 70). The sorted merge element reads from its several inputs the next element according to the sort order and passes it to its output (still paragraph 70). The "sorted merge" element seems to be a gather element in accordance with figure 2C, even though this is not expressly disclosed in the application.

2.3 The application explains that a naive implementation of a sorted merge element may block (see paragraph 71). If, for example, there is no pending input at an input 1 of a sorted merge element, it may be impossible to pass through any work element pending at a different

input because a late arrival at input 1 might take priority in the defined sort order. The sorted merge might thus be blocked until an "end-of-flow" indicator at input 1 (see paragraphs 71 and 72).

2.4 To reduce such blocking behaviour, a "sort value indicator" is sent along with the outputs of the hash partitioning elements "to indicate a value in the sort order that has been reached by at least one of its outputs" (see paragraph 73). This indicator "signals a downstream component that no work element with an earlier value n the sort order will be provided over this link" (*loc. cit.*).

Clarity, Article 84 EPC 1973, and claim construction

3. As explained above, the main problem addressed by the claimed solution is to avoid the described blocking behaviour of the downstream graph components, the central element of the solution being the sort value indicator.

3.1 To serve its purpose, the sort value indicator must relate to the sort order to be produced, i.e. the second key (see claim 1, line 7). Accordingly, the mention of the first key in line 17 of both requests is obviously an inaccuracy, the second key being intended instead. During the oral proceedings, the appellant confirmed this interpretation.

3.2 In broad terms, it is clear how the sort value indicator is meant to be processed by the downstream components. If a sort value indicator on any "first flow" is n, then work elements with a second key smaller than n may be received on other flows

irrespective of whether there is a work element available on the first flow (see paragraph 73).

3.3 However, the details are not specified in the claims. It is only specified that a decision to pass a work element to the "merged outputs" is made "based on the sort value indicator", but not how that decision is made.

3.3.1 In this regard the claimed "sorted merge element" must still block in certain situations. For instance, when a sorted merge element has two inputs where the keys of the work elements received at input 1 are 1, 5, 7 and those at input 2 are 4, 6, 8. When element "1" has already been processed, element "4" has arrived at input 2, but no element is waiting at input 1, the sorted merge element would have to know whether the next element at input 1 will be smaller or larger than "4" (say, a "3" or a "5") in order to decide whether element "4" can be passed through at that point in time.

3.3.2 Because it is not expressly claimed how the sort value indicator is processed, it is all the more important for the clarity of the claims that the sort value indicator itself be properly specified.

3.4 However, the claim language is also deficient, in respect to how and when the sort value indicator is computed. More specifically, the following problems exist.

3.4.1 Firstly, the sort value indicator expressing an invariant about the "subsequent work elements" on a given flow may have to be recomputed at least whenever an element is received from that flow. This is

confirmed by the appellant's own example submitted during the oral proceedings (and attached to the minutes of the oral proceedings) which shows the state of an inter-component link at two points in time with different sort indicator values. However, the fact that no recomputation is presently claimed constitutes a lack of an essential feature, Article 84 EPC 1973.

3.4.2 Secondly, the claims specify that the work elements provided to the "inter-component link" are sorted according to the first key (claim 1, line 5) but leave open whether they are also sorted by the second key. In the appellant's example, they are, with one partition element producing the work elements (value1, 2; value1, 6; value1, 8). If sorting by the second key were not a requirement, that partition element could also produce the values (value1, 2; value1, 8; value1, 6). For ease of reference, these examples are referred to below as V268 and V286, respectively.

3.5 During the oral proceedings, the appellant insisted that the input need not be sorted according to the second key, i.e. that the invention was meant to be able to deal with a situation such as V286. This assumption is not consistent with the description which discloses that a partial sort order is maintained by the sorted merge element.

3.6 However, if, for the sake of argument, the appellant's interpretation is accepted, how are the values arriving in the V286 order processed? Once the work element with second key "2" is received - and thus those with key "8" and "6" remain to be received - the pertinent sort value indicator would presumably have to be "6". If it were "8", then a "7" arriving at a different flow would be received before any further element from the V286

flow was received, and in particular before the "6". The desired order would thus be violated. This, however, would mean that the sort value indicator would have to be the minimal value ever to occur on the pertinent flow. Notably, if the work on a flow were sorted by the second key, the sort value indicator would only have to be the value of the second key of the next work element on that flow. If they are not sorted, the sort value indicator could only be generated once all second keys ever to occur on that flow are known - i.e. processed. This, however, is in tension with the idea that the computation graph is meant to be processed in parallel. Additionally, even if the work elements "occurred" on a flow in the V286 order (in particular, "8" before "6") they would still have to in the V286 situation, the values had to be passed to the merged outputs in the correct order (in particular "6" before "8"). That is, the claimed sorted merge element would not only have to wait for a "next" element to arrive on a flow but might have to wait for several such elements and select from them.

3.7 The appellant did not, during the oral proceedings, explain without any doubt which was the intended method for computing the sort value indicator and which assumptions had to be made so that the computation could be carried out. Nor did it file any clarifying amendments.

3.8 Weighing up the above considerations and in view of the disclosure in paragraph 70, the board considers that the intended interpretation is that the work elements on all flows are sorted by the second key and that the sort value indicator is meant to maintain that sorting.

3.9 However, as this essential feature is not in claim 1 of either request, claim 1 of both requests is unclear, Article 84 EPC 1973.

4. In its preliminary opinion, the board raised further clarity objections and issues relating to whether and under what conditions avoiding the mentioned blocking behaviour could be accepted as solving a technical problem. In the latter regard, it suggested that no such technical effect would be established without the claims specifying the execution on a parallel hardware (see point 5 of the summons).

4.1 In response to that objection, claim 1 of both requests was amended to include the feature that "different instances of the parallel partition elements and/or different instances of the parallel sorted merge elements are configured to be executed by respective different coprocessors of the computing system". As the basis for this amendment, paragraph 77 of the description was referred to (see the letter of 4 March 2019, page 2, paragraphs 4 and 5 from the bottom). However, paragraph 77 only makes the general statement that "The software may be implemented in a distributed manner in which different parts of the computation specified by the software are performed by different computers". The board doubts whether this general statement is sufficient basis for the more specific feature included in claim 1 of both requests.

4.2 That said, because the above deficiency under Article 84 EPC 1973 could not be resolved during the oral proceedings, the questions regarding original disclosure of the added feature or, in more general terms, the question of whether a technical effect can be ascribed to the claimed subject matter or why, were

not discussed during oral proceedings and need not be decided.

Order

For these reasons it is decided that:

The appeal is dismissed.

The Registrar: The Chairman:

N. Schneider W. Sekretaruk

Decision electronically authenticated