BESCHWERDEKAMMERN DES EUROPÄISCHEN PATENTAMTS

BOARDS OF APPEAL OF THE EUROPEAN PATENT OFFICE

CHAMBRES DE RECOURS DE L'OFFICE EUROPEEN DES BREVETS

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

Datasheet for the decision of 3 December 2010

Case Number: T 1946/07 - 3.5.02

Application Number: 01304531.5

Publication Number: 1158682

IPC: H03M 13/39

Language of the proceedings: EN

Title of invention:
Methods and apparatus for decoding of general codes on probability dependency graphs

Applicant:
Agere Systems Guardian Corporation

Opponent:
-

Headword:
-

Relevant legal provisions:
EPC Art. 56

Relevant legal provisions (EPC 1973):
-

Keyword:
"Inventive step - no all requests"

Decisions cited:
-

Catchword:
-

Europäisches Patentamt | European Patent Office | Office européen des brevets

Beschwerdekammern | Boards of Appeal | Chambres de recours

Case Number: T 1946/07 **-** 3.5.02

D E C I S I O N
of the Technical Board of Appeal 3.5.02
of 3 December 2010

Appellant: Agere Systems Guardian Corporation
9333 S John Young Parkway
Room 301E1211
Orlando
Florida 32819 (US)

Representative: Williams, David John
Page White & Farrer
Bedford House
John Street
London WC1N 2BF (GB)

Decision under appeal: **Decision of the Examining Division of the European Patent Office posted 10 May 2007 refusing European application No. 01304531.5 pursuant to Article 97(1) EPC 1973.**

Composition of the Board:

Chairman: M. Ruggiu
Members: M. Rognoni
P. Mühlens

Summary of Facts and Submissions

I. The appellant (applicant) appealed against the decision of the examining division refusing European patent application No. 01 304 531.5.

II. In the contested decision, the examining division held, *inter alia*, that the apparatus claim 1 and the method claim 2 lacked novelty with regard to the following document:

D1: Hagenauer J. *et al.*: *"Decoding and Equalization with Analog Non-linear Networks"*, European Transactions on Telecommunications, Wiley & Sons, GB, vol. 10, no. 6, November - December 1999, pages 659 - 680, XP002246822.

III. With the statement of grounds of appeal, the appellant submitted two sets of claims by way of main request and auxiliary request, whereby the main request corresponded to the claims upon which the contested decision was based. In the auxiliary request, claims 1 and 2 were amended to incorporate the limitations of dependent claims 5 and 6.

IV. In a communication dated 20 September 2010 accompanying the summons to oral proceedings, the Board introduced the following document into the appeal proceedings:

D7: N. Wiberg, *"Codes and Decoding on General Graphs"*, Linköping Studies in Science and Technology, Dissertation No. 440, 1996, Department of Electrical Engineering, Linköping University, Linköping, Sweden, XP002944061.

The Board essentially agreed with the examining division's findings that the subject-matter of claim 1 of the main request was not new with respect to D1 and furthermore expressed the view that also D7 appeared to anticipate the combination of features recited in claims 1 and 2. As to the auxiliary request, the subject-matter of claim 1 did not appear to go beyond what was already known from D7.

V. In response to the Board's communication, the appellant filed with a letter dated 3 November 2010 a main request and first to third auxiliary requests.

VI. With a letter dated 1 December 2010, the Board was advised that the appellant would not be represented at the oral proceedings scheduled for 3 December 2010.

VII. Oral proceedings in the absence of the appellant were held as scheduled on 3 December 2010.

VIII. The appellant requested in writing that the decision under appeal be set aside and that a patent be granted on the basis of the main request or on the basis of one of the first to third auxiliary requests, all filed with the letter of 3 November 2010.

IX. Claims 1 and 2 of the main request read as follows:

"1. An apparatus for decoding received bits or symbols, comprising:

a set of interconnected processing nodes implemented in the form of a probability dependency graph defined by a code used to encode the bits or

symbols, the processing nodes being arranged to implement a block-parallel decoding process (200, 700, 800) for a given block of the bits or symbols to be decoded;

the set of interconnected processing nodes comprising a plurality of check nodes (400, 600) and a plurality of variable nodes (300, 500), each of the variable nodes being associated with a corresponding one of the bits or symbols to be decoded;

wherein the given block of the received bits or symbols to be decoded is a block of *N* received bits or symbols to be decoded, and the plurality of variable nodes comprises *N* variable nodes, one for each of the *N* received bits or symbols of the given block, such that the *N* variable nodes are associated with respective ones of the *N* received bits or symbols to be decoded;

wherein the block-parallel decoding process includes:

(i) calculating at each of the check nodes a function of input from the variable nodes connected to that check node, and sending to the connected variable nodes the result of these check node calculations, wherein the check node calculations are performed in parallel; and

(ii) calculating at each of the variable nodes a function of input from the check nodes connected to that variable node, and sending to the connected check nodes the result of these variable node calculations, wherein the variable node calculations are performed in parallel."

"2. A method for decoding received bits or symbols, the method comprising the steps of:

applying the received bits or symbols to a set of interconnected processing nodes configured in the form of a probability dependency graph defined by a code used to encode the bits or symbols; and

implementing within the processing nodes a block-parallel decoding process for a given block of the bits or symbols to be decoded;

the set of interconnected processing nodes comprising a plurality of check nodes and a plurality of variable nodes, each of the variable nodes being associated with a corresponding one of the bits or symbols to be decoded;

wherein the given block of the received bits or symbols to be decoded is a block of *N* received bits or symbols to be decoded, and the plurality of variable nodes comprises *N* variable nodes, one for each of the *N* received bits or symbols of the given block, such that the *N* variable nodes are associated with respective ones of the N received bits or symbols to be decoded;

wherein the block-parallel decoding process includes:

(i) calculating at each of the check nodes a function of input from the variable nodes connected to that check node, and sending to the connected variable nodes the result of these check node calculations, wherein the check node calculations are performed in parallel; and

(ii) calculating at each of the variable nodes a function of input from the check nodes connected to that variable node, and sending to the connected check nodes the result of these variable node calculations, wherein the variable node calculations are performed in parallel."

Claims 1 and 2 of the first auxiliary request read as follows:

"1. An apparatus for decoding received bits or symbols, comprising:

a set of interconnected processing nodes implemented in the form of a bipartite probability dependency graph defined by a code used to encode the bits or symbols, the processing nodes being arranged to implement a block-parallel decoding process (200, 700, 800) for a given block of the bits or symbols to be decoded;

the set of interconnected processing nodes comprising a set of T check nodes (400, 600) and a plurality of variable nodes (300, 500), each of the variable nodes being associated with a corresponding one of the bits or symbols to be decoded;

wherein the given block of the received bits or symbols to be decoded is a block of *N* received bits or symbols to be decoded, and the plurality of variable nodes comprises *N* variable nodes, one for each of the *N* received bits or symbols of the given block, such that the *N* variable nodes are associated with respective ones of the *N* received bits or symbols to be decoded;

wherein the block-parallel decoding process is initiated by sending to each check node connected to a variable node a function $f(Rx_i)$ where Rx_i is the received bit or symbol, $i = 0, 1, \ldots N - 1$, and $f(.)$ is a function determined by the code comprising at least one of a threshold function, a linear scaling function and an approximate exponential function, and wherein the block-parallel decoding process includes:

(i) calculating at each of the check nodes a function of input from the variable nodes connected to

that check node, and sending to the connected variable nodes the result of these check node calculations, wherein the check node calculations are performed in parallel;

(ii) calculating at each of the variable nodes a function of input from the check nodes connected to that variable node, and sending to the connected check nodes the result of these variable node calculations, wherein the variable node calculations are performed in parallel; and

(iii) calculating at each of the check nodes a function *g (.)* of the input from the variable nodes connected to that check node, and sending to the connected variable nodes the result of these check node calculations, wherein the calculations are performed in parallel, wherein the function *g (.)* comprises at least one of a parity check function, a hyperbolic tangent function in the log domain, and a maximum function in the log domain."

"2. A method for decoding received bits or symbols, the method comprising the steps of:

applying the received bits or symbols to a set of interconnected processing nodes configured in the form of a bipartite probability dependency graph defined by a code used to encode the bits or symbols;and

implementing within the processing nodes a block-parallel decoding process for a given block of the bits or symbols to be decoded;

the set of interconnected processing nodes comprising a set of T check nodes and a plurality of variable nodes, each of the variable nodes being associated with a corresponding one of the bits or symbols to be decoded;

wherein the given block of the received bits or symbols to be decoded is a block of *N* received bits or symbols to be decoded, and the plurality of variable nodes comprises *N* variable nodes, one for each of the *N* received bits or symbols of the given block, such that the *N* variable nodes are associated with respective ones of the *N* received bits or symbols to be decoded;

wherein the block-parallel decoding process is initiated by sending to each check node connected to a variable node a function $f(Rx_i)$ where Rx_i is the received bit or symbol, $i = 0, 1, \ldots N-1$, and $f(.)$ is a function determined by the code comprising at least one of a threshold function, a linear scaling function and an approximate exponential function, and

wherein the block-parallel decoding process includes:

(i) calculating at each of the check nodes a function of input from the variable nodes connected to that check node, and sending to the connected variable nodes the result of these check node calculations, wherein the check node calculations are performed in parallel;

(ii) calculating at each of the variable nodes a function of input from the check nodes connected to that variable node, and sending to the connected check nodes the result of these variable node calculations, wherein the variable node calculations are performed in parallel; and

(iii) calculating at each of the check nodes a function $g(.)$ of the input from the variable nodes connected to that check node, and sending to the connected variable nodes the result of these check node calculations, wherein the calculations are performed in parallel, wherein the function $g(.)$ comprises at least

one of a parity check function, a hyperbolic tangent function in the log domain, and a maximum function in the log domain."

Claims 1 and 2 according to the second auxiliary request differ from claims 1 and 2 of the main request in that they further comprise the following features:

"wherein the block-parallel decoding process is initiated by sending to each check node connected to a variable node a function $f(Rx_i)$ where Rx_i is the received bit or symbol, $i = 0, 1, \ldots N-1$, and $f(.)$ is a function determined at least in part by the code; and

wherein the function f (.) comprises at least one of a threshold function, a linear scaling function and an approximate exponential function."

Claim 1 of the third auxiliary request differs from claim 1 of the second auxiliary request in that a set of interconnected processing nodes is implemented in the form of "a bipartite probability dependency graph" and the decoding process further includes the following step:

"(iii) calculating at each of the check nodes a function $g(.)$ of the input from the variable nodes connected to that check node, and sending to the connected variable nodes the result of these check node calculations, wherein the calculations are performed in parallel, wherein the function $g(.)$ comprises at least one of a parity check function, a hyperbolic tangent function in the log domain, and a maximum function in the log domain."

Claim 2 of the third auxiliary request is the same as claim 2 of the second auxiliary request.

X. With the letter dated 3 November 2010, the appellant made essentially the following submissions:

Main request

Claim 1, in addition to calling for a number N of variable nodes to be decoded, expressly called for each of the variable nodes to perform certain computations in a block-parallel decoding process. More specifically, each of the variable nodes was configured to calculate a function of the input from the check nodes connected to that variable nodes, and to send a result of its variable node calculations to the connected check nodes. The arrangements from D1 and D7 did not operate in the recited manner.

The analogue decoder network in Figure 15 of D1 clearly included only four variable nodes, denoted by circles, one for each of the information bits u_1, u_2, u_3 and u_4. One of these variable nodes was initialised by performing a computation. The others were set to zero. As the variable nodes were not described as performing computations, other than the initialisation computation mentioned above, the analogue decoder network of Figure 15 of D1 did not itself meet the limitations of claim 1.

It was further noted that the independent claims now included limitations which indicated that the particular recited variable node and check note

calculations were performed in parallel. Thus, the calculations associated with all N of the variable nodes were performed in parallel. Figure 3.4 on page 16 of D7 seemed to indicate that the calculations for all N of the variable nodes were not performed simultaneously, since the calculation for a site *s* on the right side of the figure clearly required inputs from other sites identified as s_1 and s_2. The fact that D7 on page 12 referred to parallel processing algorithm and on page 21 indicated that check-to-site cost functions would be updated simultaneously, did not necessarily indicate that the recited calculations performed by each of the N variable nodes were performed in parallel, or that the recited calculations performed by each of the check nodes were performed in parallel.

Claims 1 and 2 of the main request were specifically directed to the novel decoder structure of the present application as shown in Figure 2 of the drawings and described in the patent specification at page 9, line 13, to page 10, line 11. Such a decoder structure was not taught or suggested by D1 or D7.

First auxiliary request

The first auxiliary request incorporated dependent claims 3 to 8 into each of claims 1 and 2. Such an amendment was intended to have the claims more closely track the novel arrangement of Figure 2.

Second auxiliary request

The second auxiliary request corresponded to the previous only auxiliary request. The primary limitation at issue related to initiation of the recited block-parallel decoding process by sending a function of a received bit or symbol to each check node connected to the corresponding variable node for that bit of symbol. D7 failed to teach or suggest the recited initialisation.

One of the two limitations included in the independent claims specified that the initiation required sending a function of a received bit or symbol to each check nodes connected to the corresponding variable node for that bit or symbol. This specified recitation was clearly not met by the general statement regarding initializing local cost functions in D7. For example there was no description whatsoever regarding initialisation by communication of functions of a received bit or symbol to the check nodes connected to the site corresponding to that received bit or symbol. Accordingly, the initialisation limitation in question was not met by the teachings in D7.

Third auxiliary request

The third auxiliary further amended claim 1 of the second auxiliary request to incorporate dependent claims 3 to 6 into claim 1. Such an amendment was intended to have the claim more closely track the novel arrangement in Figure 2.

In summary, the advantageous decoder structure illustrated in Figure 2 and covered by the present claims was believed to be with novel and inventive in view of the decoder structure disclosed in Figure 15 of D1 or Figure 3.4 of D7.

Reasons for the decision

1. The appeal is admissible.

2.1 The present application is concerned with providing "*block-parallel decoding algorithms and corresponding decoder architectures for performing iterative decoding of a wide variety of different codes, including for example, convolutional codes, Turbo codes or other concatenated codes which are based on convolutional codes, block codes, etc.*" (published application, paragraph [0015]).

2.2 Accordingly, "*a block-parallel decoding algorithm and corresponding decoder architecture utilizes a set of interconnected processing nodes configured in the form of a probability dependency graph*" (paragraph [0017], first sentence - underlining added).

As pointed out in the description (paragraph [0027]), the term *"probability dependency graph"* is intended to include *"any representation of interconnected nodes configured such that an exact or approximate probability of a value associated with a given node can be expressed in terms of exact or approximate probabilities of one or more values associated with one or more nodes connected to the given node"*.

Main Request

3. Claim 2 according to the main request relates to a *"method for decoding received bits or symbols"* comprising the following steps and features:

(a) applying the received bits or symbols to a set of interconnected processing nodes configured in the form of a probability dependency graph defined by a code used to encode the bits or symbols; and

(b) implementing within the processing nodes a block-parallel decoding process for a given block of the bits or symbols to be decoded;

(c) the set of interconnected processing nodes comprising a plurality of check nodes and a plurality of variable nodes, each of the variable nodes being associated with a corresponding one of the bits or symbols to be decoded;

(d) wherein the given block of the received bits or symbols to be decoded is a block of N received bits or symbols to be decoded; and

(e) the plurality of variable nodes comprises N variable nodes, one for each of the N received bits or symbols of the given block, such that the N variable nodes are associated with respective ones of the N received bits or symbols to be decoded;

wherein the block-parallel decoding process includes:

(f_1) calculating at each of the check nodes a function of input from the variable nodes connected to that check node, and sending to the connected variable nodes the result of these check node calculations, wherein

(f_2) the check node calculations are performed in parallel; and

(g_1) calculating at each of the variable nodes a function of input from the check nodes connected to that variable node, and sending to the connected check nodes the result of these variable node calculations, wherein

(g_2) the variable node calculations are performed in parallel.

4.1 D7 is, *inter alia*, concerned with the problem of decoding codes which can be described by means of "equation systems". As pointed out on page 6, first paragraph, the equation system describing a code defines a "*bipartite graph*" with vertices both for the variables and for the equations, while an edge indicates that a particular variable is present in a particular equation. A bipartite graph used to state constraints or equations which specify error correcting codes is known as *Tanner graph* (cf published application, paragraph [0052]).

4.2 D7 (page 12, first paragraph) describes "*two generic decoding algorithms for code realizations based on Tanner graphs, ...The structure of the algorithms matches the graph directly. It will be convenient to think of these algorithms as parallel processing algorithms, where each site and each check is assigned its own processor and the communication between them reflects the Tanner graph. (In fact, this "distributed" viewpoint was one of the motivations for developing the framework. However, in many cases a sequential implementation is actually more natural.)*" (Underlining added)

As shown in Figure 3.1 (page 13) of D7 the algorithms compute a set of *"final cost functions"* μ_s for the sites *s* upon which final decisions can be made. The channel output enters the algorithms as a set of *"local cost functions* γ_s*"* for the sites *s* and as a set of *"local cost functions* γ_E*"* for the checks *E* (see D7, page 12, last paragraph).

4.3 The *"min-sum algorithm"* described in chapter 3.1 of D7 (pages 16 and 17) consists of the following steps:

- *Initialization:* The local cost functions γ_s and γ_E are initialized as appropriate (using, e.g., channel information). The intermediate site-to-check cost functions $\mu_{s,E}$ and check-to-site cost functions $\mu_{E,s}$ are set to zero.

- *Iteration:* The intermediate cost functions $\mu_{s,E}$ and $\mu_{E,s}$ are alternately updated a suitable number of times. The site-to-check cost $\mu_{s,E}(a)$ is computed as the sum of the site's local cost and all

contributions coming into *s* except the one from *E*, according to formula (3.1) shown on page 16. The check-to-site cost $\mu_{E,s}(a)$ is obtained by examining all locally valid configurations on *E* that match *a* on the site *s* and, for each configuration, summing the check's local cost and all contributions coming into *E* except the one from *s*. The minimum over these sums is taken as the cost $\mu_{E,s}(a)$ (see equation (3.2) on page 16).

- *Termination:* The final cost function $\mu_s(a)$ is computed as the sum of the site's local cost and all contributions coming into *s*. The final check cost $\mu_E(a)$ for a local configuration (a) is computed as the sum of the check's local cost and all contributions coming into E.

The goal of the "min-sum algorithm" is to find a configuration with the smallest sum of final site and check costs. As pointed out on page 17 of D7 (fourth paragraph), in a typical channel-decoding situation where the local check costs are set to zero and the local site costs are expressed in term of log-likelihoods, *"the global cost G(x) becomes the log-likelihood - log p(y|x) for the codeword x"*. The maximum-likelihood decoding corresponds to finding a valid codeword *x* that minimizes *G(x)*.

4.5 According to D7, page 18, fourth paragraph, the "*sum - product algorithm is a straightforward generalization of the forward-backward algorithm of Bahl et al [17] for the computation of per-symbol a posteriori probabilities in a trellis. Two other special cases of the sum-product algorithm are the classic turbo decoder*

by Berrou et al. [7] and Gallager's decoding algorithms for low-density parity-check codes [5]" (cf. published application [0052]).

4.6 In summary, D7 discloses a method for decoding received bits or symbols which comprises steps (a) to (e), (f_1) and (g_1) recited in claim 2 of the main request.

5.1 As to steps (f_2) and (g_2), the appellant has essentially argued, that Figure 3.4 on page 16 of D7 seemed to indicate that the calculations for all N of the variable nodes were not performed simultaneously, since the calculations for a site *s* on the right side of the figure clearly required inputs from other sites identified as s_1 and s_2. Thus, the fact that D7 on page 12 referred to *"parallel processing algorithms"* and on page 21 indicated that *"check-to-site cost functions would be updated simultaneously"* did not necessarily indicate that the recited calculations performed by each of the *N* variable nodes were performed in parallel, or that the recited calculations performed by each of the check nodes were performed in parallel.

5.2 Figure 3.3 a) of D7, page 14, shows an example of Tanner graph. The circles (sites) correspond to the codeword components and the black dots (checks) to the parity check equations. This means that in the given example the three sites connected to any of the three checks are required to have even parity.

Figure 3.4 on page 16 illustrates the updating rules for the site-to-check costs and the check-to-site costs of the "min-sum algorithm". According to D7, page 16

(see *"Iteration"*), the intermediate cost functions $\mu_{s,E}$ and $\mu_{E,s}$ are *"alternatingly updated a suitable number of times"*. As pointed out above, the site-to-check costs $\mu_{s,E}$ and the check-to-site costs $\mu_{E,s}$ are functions of the corresponding local cost functions γ_s and γ_E and of the check-to site costs $\mu_{E,s}$ and site-to-check costs $\mu_{s,E}$, respectively.

5.3 The algorithm disclosed in D7 and summarized above corresponds essentially to the one described in paragraph [0032] of the application as follows:

- *"The decoder 200 is initialized by sending to each check node connected to a variable node a function of the received value, e.g., f (Rx_i) where Rx_i is the received bit or symbol, i = 0, 1, ... N-1, and f (.) is a function determined by the code, e.g., a threshold function, linear scaling function, or approximate exponential function. This initialization is shown on the left side of the variable nodes in FIG. 2. The decoder 200 then iterates the following operations:*

  1. At each of the check nodes, denoted C_0, C_1, ... C_{T-1}, calculate a possibly different function g (.) of the input from the variable nodes connected to that check node. For example, different functions g (.) may be used at different check nodes, or the function(s) used could change with the number of iterations or another parameter. The function g (.) used at a given one of the check nodes may be, e.g., a parity check function, a hyperbolic tangent function in the log domain,

or a maximum function in the log domain. Send to the connected variable nodes the result of these calculations, which is usually a different value for each connected variable node. These calculations may be performed in parallel.

2. At each of the variable nodes, denoted V_0, V_1, ... V_{N-1}, calculate a possibly different function h(.) of the input from the check nodes connected to that variable node. For example, different functions h (.) may be used at different variable nodes, or the function(s) used could change with the number of iterations or another parameter. The function h(.) used at a given one of the variable nodes may include, e.g., a majority function or an averaging function. Send to the connected check nodes the result of these calculations. Again, these calculations may be performed in parallel."

5.4 In other words, both the present application and D7 teach to perform iterations by updating alternatively the cost functions of the sites (variable nodes) and of the check nodes for an initially received codeword.

Even if in the examples shown in Figures 3.3 and 3.4 of D7 it is not specified that the calculations for all check nodes and the calculations for all site nodes are to be performed in parallel, D7 teaches explicitly on page 21, section 3.3, first paragraph, that it is convenient to view the algorithms as "parallel", with one processing unit for each site and each check. *"Then*

all check-to-site cost functions would be updated simultaneously, and similarly for the site-to-check cost functions. This updating order is used in Gallager's sum-product decoding algorithm for low density parity-check codes". As known in the art, Gallager's codes are typically represented by Tanner graphs of the kind shown in Figure 2 of the application.

5.5 For the sake of completeness, it should also be mentioned that according to D7 (see page 21 chapter 3.3, second paragraph) it is possible to find efficient updating orders for any cycle free check structure by selecting any site as "root" site and starting *"updating the cost functions at the leaves, working towards the root site, but always finishing all the subgraphs leading into a check or site before proceeding with its outgoing cost function. For each site and check, only the cost function pointing towards the root is updated."* Although for this kind of graph it is evidently not possible to calculate all variable nodes and all check nodes in parallel, it is implicit that all the "subgraphs" leading into a check node or a variable node should be updated simultaneously.

In fact, the present application also shows that parallel processing cannot always be applied to all nodes (see Figure 7 and paragraphs [0043] to [0045]). In the case of the Bayesian network probability dependency graph according to Figure 7, which has different levels of nodes, whereby any node has inputs only from the previous level and outputs to the subsequent level, parallel processing can evidently only be performed within a particular level.

5.6 In summary, the Board finds that it would be obvious to a person skilled in the art, starting from the teaching of D7 and wishing to decode *e.g.* a code described by a set of equations, to arrive at a method falling within the terms of claim 2 of the present application.

6.1 Hence, the subject-matter of claim 2 does not involve an inventive step within the meaning of Article 56 EPC.

6.2 The same applies, *mutatis mutandis,* to the subject-matter of claim 1 which relates to an apparatus essentially comprising means for carrying out the method of claim 1.

First auxiliary request

7.1 Claim 2 according to the first auxiliary request differs from claim 2 of the main request in that the processing nodes are configured in the form of a *"bipartite"* probability dependency graph, and in that it further comprises the following features relating to the initiation of the decoding process:

(h) wherein the block-parallel decoding process is initiated by sending to each check node connected to a variable node a function $f\ (Rx_i)$ where Rx_i is the received bit or symbol, $i = 0, 1, \ldots N - 1$, and

(j) $f\ (.)$ is a function determined by the code comprising at least one of a threshold function, a linear scaling function and an approximate exponential function,

and the block-parallel decoding process includes the following step:

(iii) calculating at each of the check nodes a function *g (.)* of the input from the variable nodes connected to that check node, and sending to the connected variable nodes the result of these check node calculations, wherein the calculations are performed in parallel, wherein the function *g (.)* comprises at least one of a parity check function, a hyperbolic tangent function in the log domain, and a maximum function in the log domain.

7.2 As pointed out above (see item 4.1), D7 is concerned with codes represented by *"bipartite graphs"* such as Tanner graphs.

7.3 As to feature (iii), it is explained on page 14 of D7 (see Figure 3.3a) that the check nodes correspond to the parity check equations. In the example of Figure 3.3a, they express the condition that all the sites connected to a check node are required to have even parity. The check-to-site cost function $\mu_{E,s}$ represents all the possible cost contributions from the check *E* to the site *s* which satisfy the parity check condition. The minimum cost contribution is actually sent to the node *s*. As the check cost contributions are calculated only for bit combinations that satisfy the parity check function, the check-to-site cost functions are related to the parity check function (see feature (iii) above.

7.4 As to features (h) and (j), it is specified in D7 (page 6), that the local cost functions γ_s and γ_E are initialized as appropriate (using, e. g., channel information), whereas the intermediate cost functions $\mu_{s,E}$ and $\mu_{E,s}$ are set to zero. As pointed out on page 16, first paragraph, when *"using the min-sum algorithm in a channel-decoding situation with a memoryless channel and a received vector y, the local check costs $\gamma_E(x_E)$ are typically omitted (set to zero) and the local site costs $\gamma_s(x_s)$ are the usual channel log-likelihoods - log $p(y_s|x_s)$".*

This implies that the decoding process is initiated by sending to each check node connected to a variable node a function f (Rx_i) where Rx_i is the received bit or symbol, i = 0, 1, ... N-1, and f (.) represents the log-likelihood - log $p(y_s|x_s)$ of the received symbol y_s. As known in the art (see D1, section 3.3), log-likelihood ratios can be expressed in terms of exponential functions of the received symbol amplitude.

7.5 Furthermore Figures 3 and 4 of the present application show two examples of variable and check nodes, respectively, in a hard-decision implementation. Figures 5 and 6 show examples of variable and check nodes, respectively, for use in a soft-decision decoder. According to paragraph [0040] of the published application, the received value Rx_i in a soft-decision implementation of the decoder of Figure 5 is operated on by a function f(.), e. g. a linear scaling function. In fact, the threshold function, linear scaling function or approximate exponential function referred to in claim 2 appear to be the functions which are required to quantize the received analogue values or to

scale them prior to hard-decision or soft-decision decoding.

7.6 In summary, the Board considers that it would be obvious to a person skilled in the art wishing to implement the teaching of D7 to carry out a hard-decision or soft-decision decoding process on input variables representing a threshold function, a linear scaling function or an approximate exponential function of the received analogue values. In doing so, the skilled person would arrive at a method falling within the terms of claim 2 of the first auxiliary request.

8.1 Hence, the subject-matter of claim 2 does not involve an inventive step within the meaning of Article 56 EPC.

8.2 The same applies, *mutatis mutandis,* to the subject-matter of claim 1 which relates to an apparatus essentially comprising means for carrying out the method of claim 1.

Second auxiliary request

10. Claim 2 according to the second auxiliary request differs from claim 1 of the main request essentially in that it further comprises features (h) and (j) (see item 7.1).

11.1 For the reasons given above (see items 7.4 to 7.6). The subject-matter of claim 2 does not involve an inventive step within the meaning of Article 56 EPC.

11.2 The same applies, *mutatis mutandis,* to the subject-matter of claim 1 which relates to an apparatus

essentially comprising means for carrying out the method of claim 1.

Third auxiliary request

12. Claim 2 of the third auxiliary is the same as claim 2 of the second auxiliary request. Thus, for the reasons given above, it does not satisfy the requirements of Article 56 EPC.

13. As to claim 1, it differs from claim 1 of the second auxiliary request in that the processing nodes are implemented in the form of a *"bipartite"* probability dependency graph and the block-parallel decoding process includes step (iii) (see item 7.1). For the reasons given above (see items 7.2 and 7.3), the subject-matter of this claim does not involve an inventive step.

14 In the result, the Board finds that none of the appellant's requests relates to patentable subject-matter. Hence, the application has to be refused.

Order

For the above reasons it is decided that:

The appeal is dismissed.

The Registrar: The Chairman:

U. Bultmann M. Ruggiu