generating a parity check matrix, *H*, using the selected two integer satisfying the following conditions:

$$H(row,col)=\begin{cases}0, & when\ 0\le idx<eta-1, and\\ & c\cdot(idx+1)\le col<n-(eta-idx-1)\cdot c\\ 0, & when\ eta-1\le \mathrm{idx}<\dfrac{\mathrm{n-m}}{\mathrm{c-b}}, \mathrm{and}\\ & (\ 0\le col<(idx-eta+1)\cdot c\ or\\ & (idx-eta+1)\cdot c+n_b\le col<n)\end{cases}$$

Where idx is integer, $row = idx\cdot(c-b), idx\cdot(c-b)+1,\cdots, idx\cdot(c-b)+c-b-1, eta=\gcd(m_b,n_b), c=\frac{n_b}{eta}, b=\frac{m_b}{eta}$, and encoding the data using an erasure code with the determined parity check matrix, *H*, to form encoded data and parity chunks.

In accordance with an embodiment of this invention, the step of generating a parity check matrix, *H comprises selecting a set of parity check matrices* $H_{b,t}$ *as base codes,* generating two sets of matrices $H_{u,t}$ and $H_{l,t}$ with the same dimensions (n_b -m_b) × n_b, generating a set of parity check matrices, H_{rep}, with variable dimensions (*rep* x (n_b -m_b) x (*rep* x n_b) based on the two sets of matrices $H_{u,t}$ and $H_{l,t}$.

In accordance with an embodiment of this invention, the step of selecting a set of parity check matrices $H_{b,t}$ as base codes with the dimensions (n_b -m_b) × n_b comprises generating a set of normal block codes with parity check matrices $H_{b,t}$ with the size (n_b -m_b) × n_b, where 0 ≤ t ≤ max_rep-1, and max_rep is an integer related to the maximum size of the generated parity check matrix, *H*.

In accordance with an embodiment of this invention, the step of generating two sets of matrices $H_{u,t}$ and $H_{l,t}$ with the same dimensions (n_b -m_b) × n_b comprises cutting each of the parity check matrices $H_{b,t}$ into two pieces with a cutting pattern such that $H_{b,t}$ = $H_{u,t}$ + $H_{l,t}$. Preferably, the cutting pattern is *eta* = *gcd*(m_b,n_b) and *b* = m_b/*eta*, *c* = n_b/*eta*, where c-units are moved to the right and down by c-b unit.

In accordance with an embodiment of this invention, the step of determining value of *rep1* using Fsize and Csize comprises determining the smallest number, *p*, with the following expression, p = FSize/(CSize*m_b), setting *rep1* = *p*, if *p* is an integer, and setting *rep1* = smallest integer that is bigger than *p*, if *p* is not an integer.

In accordance with an embodiment of this invention, the step of determining value of *rep2* using the FSizeNew comprises determining the smallest number, *pnew*, with the following expression, *pnew* = FSizeNew/(CSize*m_b), setting *rep2* = *pnew*, if *pnew* is an integer, and setting *rep2* = smallest integer that is bigger than *pnew*, if *pnew* is not an integer.

In accordance with an embodiment of this invention, the step of determining a new parity check matrix *Hnew* using *rep2* comprises generating two sets of matrices $H_{u,t}$ and $H_{l,t}$ with the same dimensions (n_b -m_b) × n_b, generating a set of parity check matrices, H_{rep}, with variable dimensions (*rep2* x (n_b -m_b) x (*rep2* x n_b) based on the two sets of matrices $H_{u,t}$ and $H_{l,t}$, determining the new parity check matrix, *Hnew*, from the set of parity check matrices satisfying the following conditions:

$$H(row,col)=\begin{cases}0, & when\ 0\le idx<eta-1, and\\ & c\cdot(idx+1)\le col<n-(eta-idx-1)\cdot c\\ 0, & when\ eta-1\le \mathrm{idx}<\dfrac{\mathrm{n-m}}{\mathrm{c-b}}, \mathrm{and}\\ & (\ 0\le col<(idx-eta+1)\cdot c\ or\\ & (idx-eta+1)\cdot c+n_b\le col<n)\end{cases}$$

Where idx is integer, $row = idx\cdot(c-b), idx\cdot(c-b)+1,\cdots, idx\cdot(c-b)+c-b-1, eta=\gcd(m_b,n_b), c=\frac{n_b}{eta}, b=\frac{m_b}{eta}$.

In accordance with an embodiment of this invention, the method of encoding is implemented via an application program stored on a memory of a processing unit.

Figure 5 illustrates a flow diagram of process 500 performed by the processor in processing system 400 in accordance with an embodiment of this invention. Process 500 is a process being executable upon receiving data for storing on a distributed storage system. Particularly, process 500 is a process for generating LDPC codes.

Process 500 begins with step 505 by selecting two integers n_b and m_b where $n_b/m_b > 1$ is required. The two integers n_b and m_b decide the redundancy rate. For example, if 1GB data is to be stored, at least $n_b/m_b > 1$ GB space is required. If 2GB data is to be stored, $2n_b/m_b$ has to be at least more than 2.

The parity check matrices generated should have zeroes at the following positions shown in the following Equation (1).

$$H(row,col) = \begin{cases} 0, & when\ 0 \le idx < eta - 1, and \\ & c \cdot (idx+1) \le col < n - (eta - idx - 1) \cdot c \\ 0, & when\ eta - 1 \le \text{idx} < \frac{\text{n-m}}{\text{c-b}}, \text{and} \\ & (\ 0 \le col < (idx - eta + 1) \cdot c\ or \\ & (idx - eta + 1) \cdot c + n_b \le col < n) \end{cases} \quad (1)$$

Where *idx* is integer, *eta* = $gcd(m_b, n_b)$ and $b = m_b/eta$, $c = n_b/eta$, $row = idx \cdot (c-b), idx \cdot (c-b)+1, \cdots, idx \cdot (c-b) + c - b - 1$

It is worth mentioning that for purposes of this discussion, the indices of the matrix start from 0 instead of 1.

Hence, in step 510, process 500 generates at least a parity check matrix *H* satisfying the constraints shown in Equation (1) above. To assist this step of generating a parity check matrix *H*, the following steps may be performed:

Claims:

1. An encoding method for adding redundancy to data to be stored on a computer system or a distributed storage system comprising:

receiving data to be encoded;

selecting two integers n_b and m_b, where n_b/m_b is more than 1;

generating a parity check matrix, H, using the selected n_b and m_b satisfying the following conditions:

$$H(row, col) = \begin{cases} 0, & when\ 0 \le idx < eta - 1, and \\ & c \cdot (idx + 1) \le col < n - (eta - idx - 1) \cdot c \\ 0, & when\ eta - 1 \le \mathrm{idx} < \dfrac{\mathrm{n} - \mathrm{m}}{\mathrm{c} - \mathrm{b}}, \mathrm{and} \\ & (\ 0 \le col < (idx - eta + 1) \cdot c\ or \\ & (idx - eta + 1) \cdot c + n_b \le col < n) \end{cases}$$

Where idx is integer, $row = idx \cdot (c - b), idx \cdot (c - b) + 1, \cdots, idx \cdot (c - b) + c - b - 1, eta = \gcd(m_b, n_b), c = \frac{n_b}{eta}, b = \frac{m_b}{eta}$, $n - m$ and n are the row and col of the H matrix generated, and

encoding said data using an erasure codes with the said determined parity check matrix, H, to form encoded data and parity chunks.

2. The method according to claim 1, wherein said generating a parity check matrix, H, comprises:

selecting a set of parity check matrices $H_{b,t}$ as base codes with the dimensions (n_b -m_b) × n_b

generating two sets of matrices $H_{u,t}$ and $H_{l,t}$ with the same dimensions (n_b -m_b) × n_b; and

generating a set of parity check matrices, H_{rep}, with variable dimensions (*rep* x (n_b -m_b) x (*rep* x n_b) based on the two sets of matrices $H_{u,t}$ and $H_{l,t}$.

3. The method according to claim 2, wherein said selecting a set of parity check matrices $H_{b,t}$ as base codes with the dimensions (n_b -m_b) × n_b comprises:

generating a set of normal block codes with parity check matrices $H_{b,t}$ with the size $(n_b - m_b) \times n_b$, where $0 \leq t \leq$ max_rep-1, and max_rep is an integer related to the maximum size of the generated parity check matrix, H.

4. The method according to claim 2, wherein said generating two sets of matrices $H_{u,t}$ and $H_{l,t}$ with the same dimensions $(n_b - m_b) \times n_b$ comprises:

cutting each of said parity check matrices $H_{b,t}$ into two pieces with a cutting pattern such that $H_{b,t} = H_{u,t} + H_{l,t}$.

5. The method according to claim 4 wherein said cutting pattern is $eta = gcd(m_b, n_b)$ and $b = m_b/eta$, $c = n_b/eta$, where c-units are moved to the right and down by c-b unit.

6. The method according to claim 1 further comprising:

assigning a node randomly to a first chunk of said encoded data and parity chunks;

selecting a next chunk from said encoded data and parity chunks;

determining a subsequent node having the shortest network path with respect to said node;

assigning said subsequent node having the shortest network path to said next chunk;

repeating said step of selecting a next chunk from said encoded data and parity chunks for each subsequent chunk in said encoded data and parity chunks, determining a subsequent node having the shortest network path with respect to preceding node and assigning said subsequent node having the shortest network path to said subsequent chunk; and

transmitting said chunks to respective assigned nodes.

7. The method according to claim 2 further comprising:

conducting column permutations.

8. The method according to claim 7 wherein said conducting column permutations comprises:

interlacing the columns associated with the parity bits with those associated with the data bits with the following rule that inserting (c-b) columns from $(m_b\text{-}1)^{th}$ columns of the base code $H_{b,l}$ in between every two b columns.

9. The method according to claim 8 further comprises:

receiving the size of the data to be encoded, FSize, and the chunk size of the encoded data, CSize;

determining value of *rep1* using said FSize and CSize;

determining the total size of the original data and the size of the new data to be appended to the original data, FSizeNew;

determining value of *rep2* using said FSizeNew;

determining a new parity check matrix *Hnew* using *rep2*;

re-encoding the first (n_b-m_b)-(c-b) parity chunks and the added chunks with the new parity check matrix *Hnew*; and

updating a meta data related to the appended data.

10. The method according to claim 9 wherein said step of determining value of *rep1* using Fsize and Csize comprises:

determining the number, p, with the following expression

$$p = \text{FSize}/(\text{CSize}*m_b);$$

setting *rep1* = p, if p is an integer; and

setting *rep1* equals to the smallest integer that is bigger than p, if p is not an integer.

11. The method according to claim 10 wherein said step of determining value of *rep2* using said FSizeNew comprises:

determining the number, *pnew*, with the following expression

$$pnew = \text{FSizeNew}/(\text{CSize}*m_b);$$

setting *rep2* = *pnew*, if *pnew* is an integer; and

setting *rep2* equals to the smallest integer that is bigger than *pnew*, if *pnew* is not an integer.

12. The method according to claim 11 wherein said determining a new parity check matrix *Hnew* using *rep2* comprises:

generating two sets of matrices $H_{u,t}$ and $H_{l,t}$ with the same dimensions $(n_b - m_b) \times n_b$;

generating a set of parity check matrices, H_{rep}, with variable dimensions (*rep2* x $(n_b - m_b)$ x (*rep2* x n_b) based on the two sets of matrices $H_{u,t}$ and $H_{l,t}$;

determining said new parity check matrix, *Hnew*, from said set of parity check matrices satisfying the following conditions:

$$H(row, col) = \begin{cases} 0, & when\ 0 \le idx < eta - 1, and \\ & c \cdot (idx + 1) \le col < n - (eta - idx - 1) \cdot c \\ 0, & when\ eta - 1 \le \text{idx} < \dfrac{\text{n} - \text{m}}{\text{c} - \text{b}}, \text{and} \\ & (\ 0 \le col < (idx - eta + 1) \cdot c\ \ or \\ & (idx - eta + 1) \cdot c + n_b \le col < n) \end{cases}$$

Where *idx* is integer, $row = idx \cdot (c - b), idx \cdot (c - b) + 1, \cdots, idx \cdot (c - b) + c - b - 1, eta = \gcd(m_b, n_b), c = \frac{n_b}{eta}, b = \frac{m_b}{eta}$, $n - m$ and n are the row and col of the *H* matrix generated.

13. A system for encoding a data and adding redundancy to data to be stored on a computer system or a distributed storage system comprising:

a processing unit having a processor and memory;

instructions stored on said memory executable by said processor to:

receiving data to be encoded;

selecting two integers n_b and m_b, where n_b/m_b is more than 1;

generating a parity check matrix, H, using said selected two integers n_b and m_b satisfying the following conditions:

$$H(row,col)=\begin{cases}0, & when\ 0\le idx<eta-1, and\\ & c\cdot(idx+1)\le col<n-(eta-idx-1)\cdot c\\ 0, & when\ eta-1\le \mathrm{idx}<\dfrac{\mathrm{n}-\mathrm{m}}{\mathrm{c}-\mathrm{b}}, \text{and}\\ & (\ 0\le col<(idx-eta+1)\cdot c\ or\\ & (idx-eta+1)\cdot c+n_b\le col<n)\end{cases}$$

Where idx is integer, $row=idx\cdot(c-b), idx\cdot(c-b)+1,\cdots, idx\cdot(c-b)+c-b-1, eta=\gcd(m_b,n_b), c=\frac{n_b}{eta}, b=\frac{m_b}{eta}$, $n-m$ and n are the row and col of the H matrix generated, and

encoding said data using an erasure code with the said determined parity check matrix, H, to form encoded data and parity chunks.

14. The system according to claim 13, wherein said instruction to generating a parity check matrix, H, comprises instructions to:

selecting a set of parity check matrices $H_{b,t}$ as base codes with the dimensions $(n_b - m_b) \times n_b$

generating two sets of matrices $H_{u,t}$ and $H_{l,t}$ with the same dimensions $(n_b - m_b) \times n_b$; and

generating a set of parity check matrices, H_{rep}, with variable dimensions (rep x $(n_b - m_b)$ x (rep x n_b) based on the two sets of matrices $H_{u,t}$ and $H_{l,t}$.

15. The system according to claim 14, wherein said instruction to selecting a set of parity check matrices $H_{b,t}$ as base codes with the dimensions $(n_b - m_b) \times n_b$ comprises

instruction to generating a set of normal block codes with parity check matrices $H_{b,t}$ with the size $(n_b - m_b) \times n_b$, where $0 \leq t \leq$ max_rep-1, and max_rep is an integer related to the maximum size of the generated parity check matrix, H.

16. The system according to claim 14, wherein said instruction to generating two sets of matrices $H_{u,t}$ and $H_{l,t}$ with the same dimensions $(n_b - m_b) \times n_b$ comprises instruction to cutting each of said parity check matrices $H_{b,t}$ into two pieces with a cutting pattern such that $H_{b,t} = H_{u,t} + H_{l,t}$.

17. The system according to claim 16 wherein said cutting pattern is $eta = gcd(m_b, n_b)$ and $b = m_b/eta$, $c = n_b/eta$, where c-units are moved to the right and down by c-b unit.

18. The system according to claim 13 further comprising:

instructions stored on said memory executable by said processor to:

assigning a node randomly to a first chunk of said encoded data and parity chunks;

selecting a next chunk from said encoded data and parity chunks;

determining a subsequent node having the shortest network path with respect to said node;

assigning said subsequent node having the shortest network path to said next chunk;

repeating said step of selecting a next chunk from said encoded data and parity chunks for each subsequent chunk in said encoded data and parity chunks, determining a subsequent node having the shortest network path with respect to preceding node and assigning said subsequent node having the shortest network path to said subsequent chunk; and

transmitting said chunks to respective assigned nodes.

19. The system according to claim 14 further comprising:

instructions stored on said memory executable by said processor to:

conducting column permutations.

20. The system according to claim 19 wherein said instruction to conducting column permutations comprises instruction to interlacing the columns associated with the parity bits with those associated with the data bits with the following rule that inserting (c-b) columns from $(m_b\text{-}1)^{th}$ columns of the base code $H_{b,t}$ in between every two b columns.

21. The system according to claim 20 further comprises:

instructions stored on said memory executable by said processor to:

receiving the size of the data to be encoded, FSize, and the chunk size of the encoded data, CSize;

determining value of *rep1* using said FSize and CSize;

determining the total size of the original data and the size of the new data to be appended to the original data, FSizeNew;

determining value of *rep2* using said FSizeNew;

determining a new parity check matrix *Hnew* using *rep2*;

re-encoding the first $(n_b\text{-}m_b)$-(c-b) parity chunks and the added chunks with the new parity check matrix *Hnew*; and

updating a meta data related to the appended data.

22. The system according to claim 21 wherein said instruction to determining value of *rep1* using Fsize and Csize comprises instructions to:

determining the number, p, with the following expression

p = FSize/(CSize*m_b);

setting *rep1* = p, if p is an integer; and

setting *rep1* equals to the smallest integer that is bigger than p, if p is not an integer.

23. The system according to claim 22 wherein said instruction to determining value of *rep2* using said FSizeNew comprises instructions to:

determining the number, *pnew*, with the following expression

pnew = FSizeNew/(CSize*m_b);

setting *rep2* = *pnew*, if *pnew* is an integer; and

setting *rep2* equals to the smallest integer that is bigger than *pnew*, if *pnew* is not an integer.

24. The system according to claim 23 wherein said instruction to determining a new parity check matrix *Hnew* using *rep2* comprises instructions to:

generating two sets of matrices $H_{u,t}$ and $H_{l,t}$ with the same dimensions (n_b -m_b) × n_b;

generating a set of parity check matrices, H_{rep}, with variable dimensions (*rep2* x (n_b -m_b) x (*rep2* x n_b) based on the two sets of matrices $H_{u,t}$ and $H_{l,t}$;

determining said new parity check matrix, *Hnew*, from said set of parity check matrices satisfying the following conditions:

$$H(row,col) = \begin{cases} 0, & when\ 0 \le idx < eta-1, and \\ & c\cdot(idx+1) \le col < n-(eta-idx-1)\cdot c \\ 0, & when\ eta-1 \le \mathrm{idx} < \dfrac{\mathrm{n-m}}{\mathrm{c-b}}, \mathrm{and} \\ & (\ 0 \le col < (idx-eta+1)\cdot c\ \ or \\ & (idx-eta+1)\cdot c + n_b \le col < n) \end{cases}$$

Where *idx* is an integer, $row = idx\cdot(c-b), idx\cdot(c-b)+1, \cdots, idx\cdot(c-b)+c-b-1, eta = \gcd(m_b, n_b), c = \frac{n_b}{eta}, b = \frac{m_b}{eta}, n-m$ and n are the row and col of the H matrix generated.