US 20140289363A1

(19) United States

(12) Patent Application Publication

(10) Pub. No.: US 2014/0289363 A1

(43) Pub. Date: Sep. 25, 2014

SENTINELLI et al.

(54) METHOD FOR DISTRIBUTING INFORMATION CONTENTS, CORRESPONDING DEVICE AND COMPUTER PROGRAM PRODUCT

(71) Applicant: **STMicroelectronics S.r.l.**, Agrate Brianza (IT)

(72) Inventors: **Alexandro SENTINELLI**, Milano (IT); **Andrea Lorenzo VITALI**, Santa Clara, CA (US); **Allan TASCHINI**, Bergamo (IT)

(73) Assignee: **STMicroelectronics S.r.l.**, Agrate Brianza (IT)

(21) Appl. No.: **14/222,770**

(22) Filed: **Mar. 24, 2014**

(30) **Foreign Application Priority Data**

Mar. 22, 2013 (IT) TO2013A000230

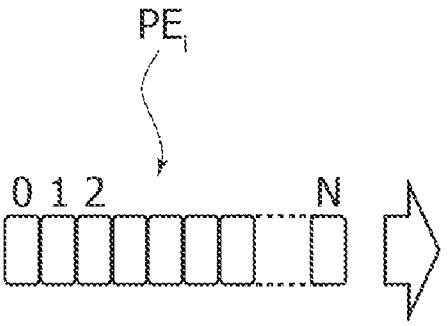
Publication Classification

(51) **Int. Cl.**
H04L 29/08 (2006.01)

(52) **U.S. Cl.**
CPC ***H04L 67/10*** (2013.01)
USPC **709/217**

(57) **ABSTRACT**

In an embodiment, information contents, such as, e.g., media contents arranged in pieces including blocks of bits, is distributed over a network including plural terminals at least one of which acts as a source of the pieces of information distributed. Various terminals in the network are configured to act as peer terminals with at least one first peer terminal sending the information to one or more second peer terminals. A set of blocks of a corresponding piece of information is received at the first peer terminal and the corresponding piece of information is reconstructed from the set of blocks received. The pieces of information distributed over the network are fountain encoded by XOR-ing the blocks in a piece, so that a received piece is reconstructable from a combination of a corresponding set of linearly independent XOR-ed blocks. The first peer terminal may start sending to the second peer terminal(s) blocks it is receiving before the corresponding piece of information is reconstructed and possibly subjected to integrity check at the first terminal. Fountain encoding the pieces of information is by means of plural sets of linearly independent XOR-ed blocks. A same piece of information is distributed to a plurality of peer terminals by sending thereto different sets of linearly independent XOR-ed blocks, thus avoiding undue redundancy in the information further propagated to other peers.



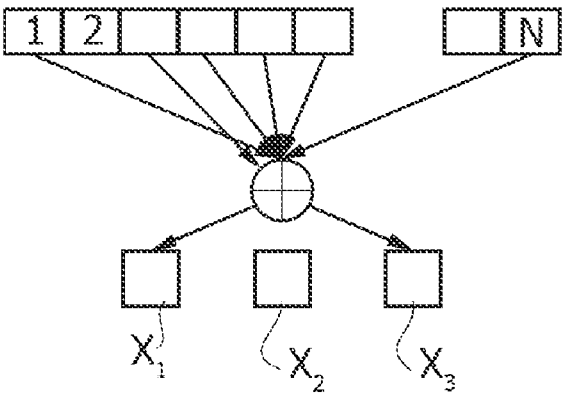


FIG. 1

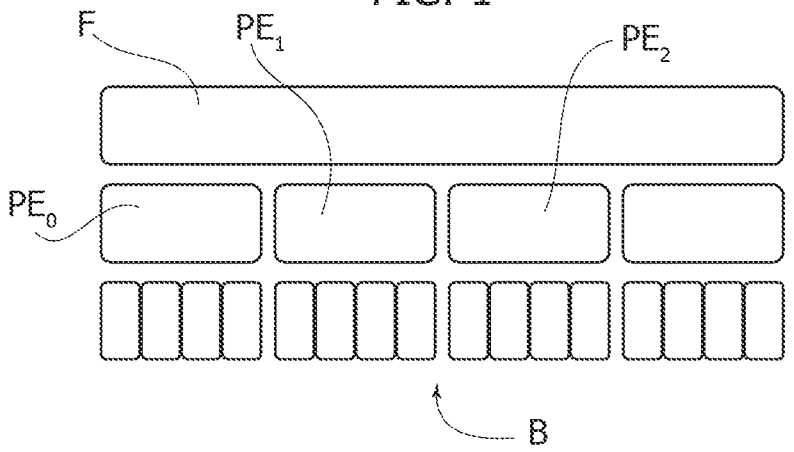


FIG. 2

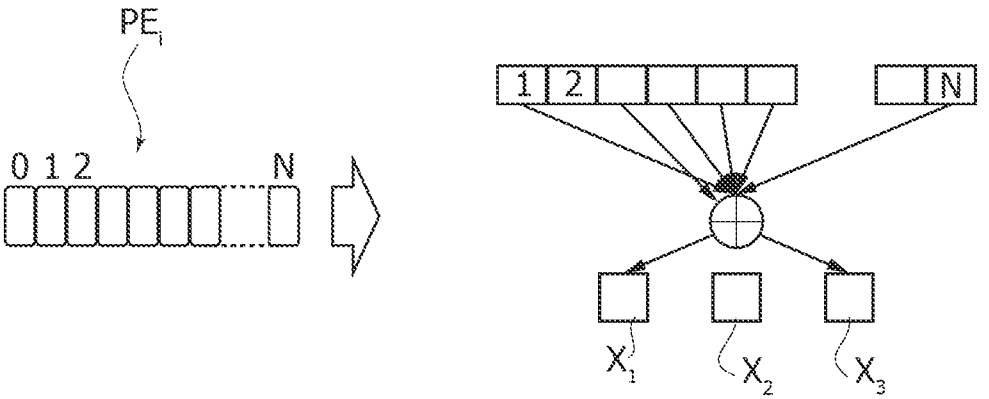
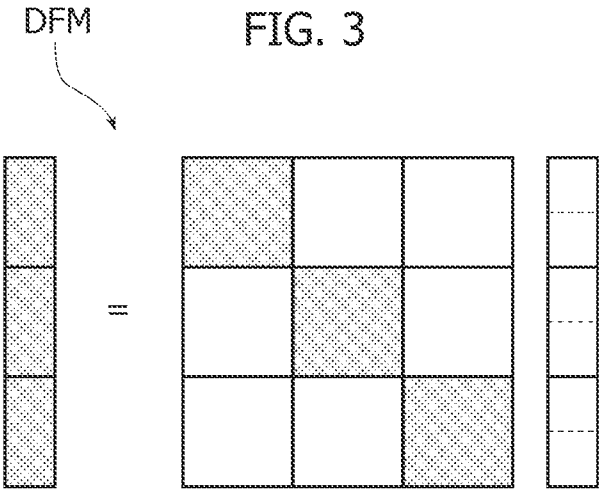


FIG. 3



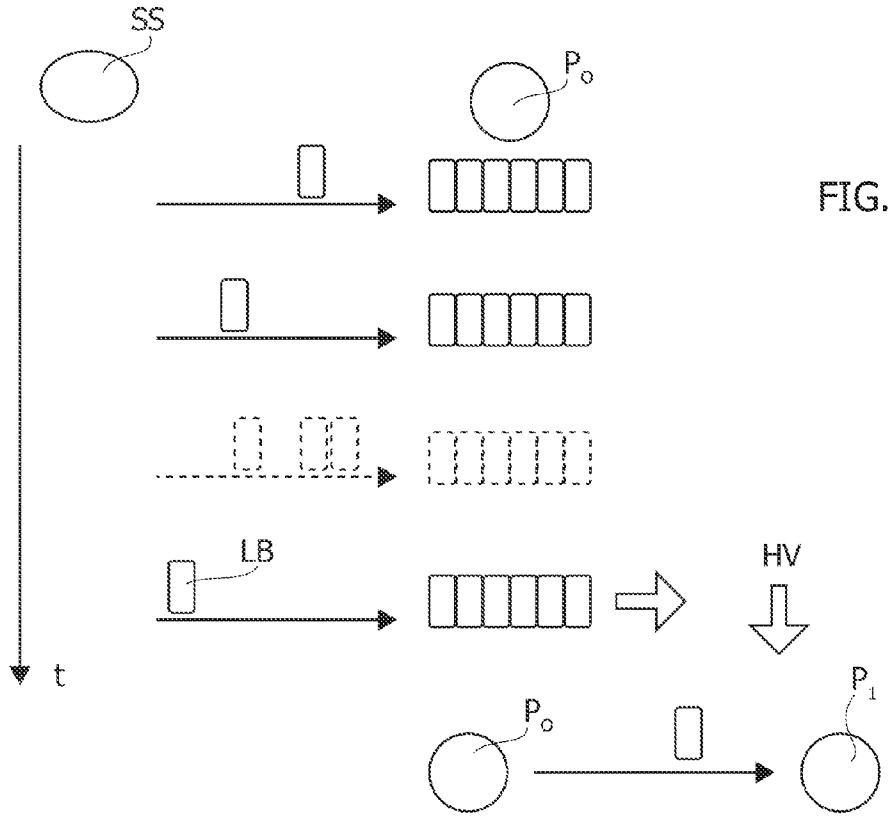


FIG. 4

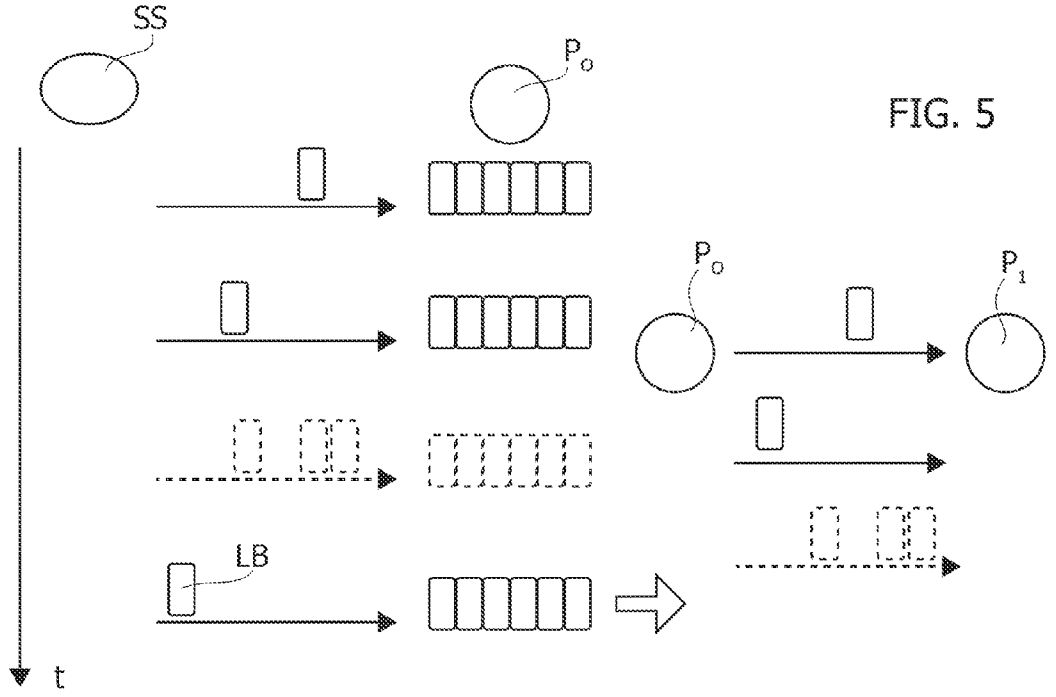


FIG. 5

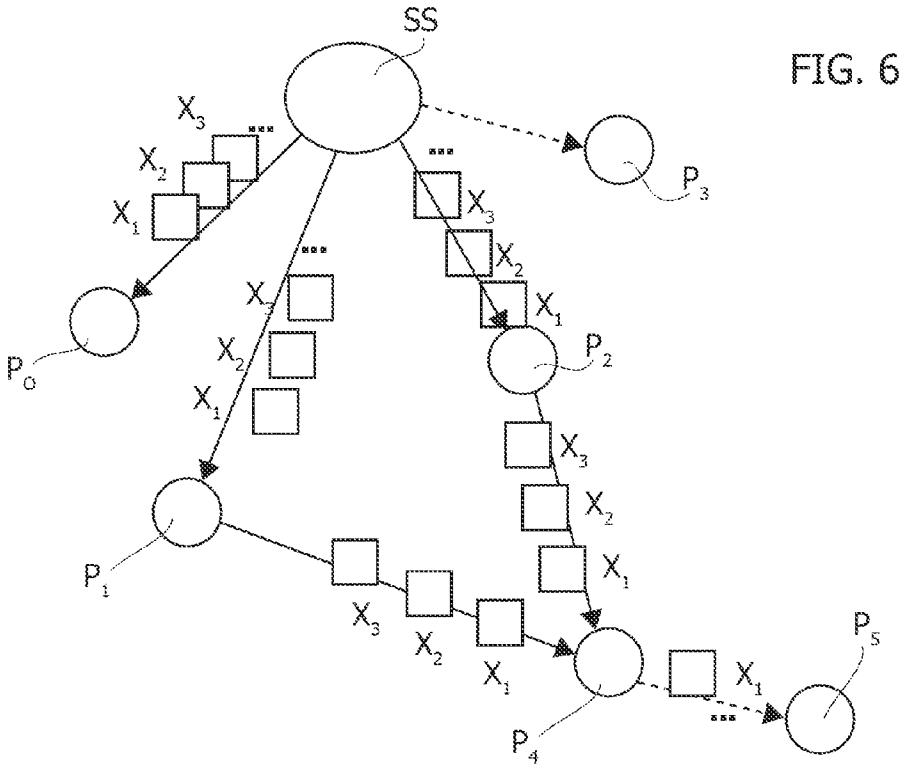


FIG. 6

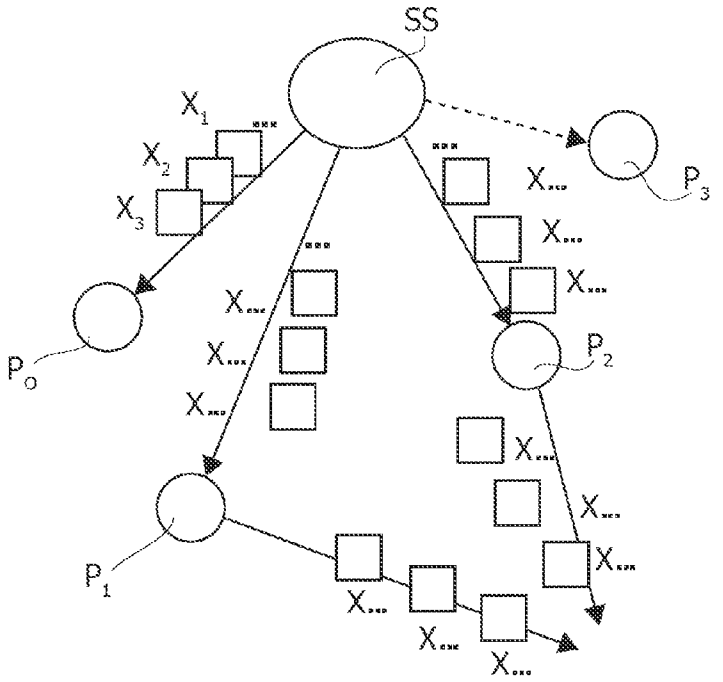


FIG. 7

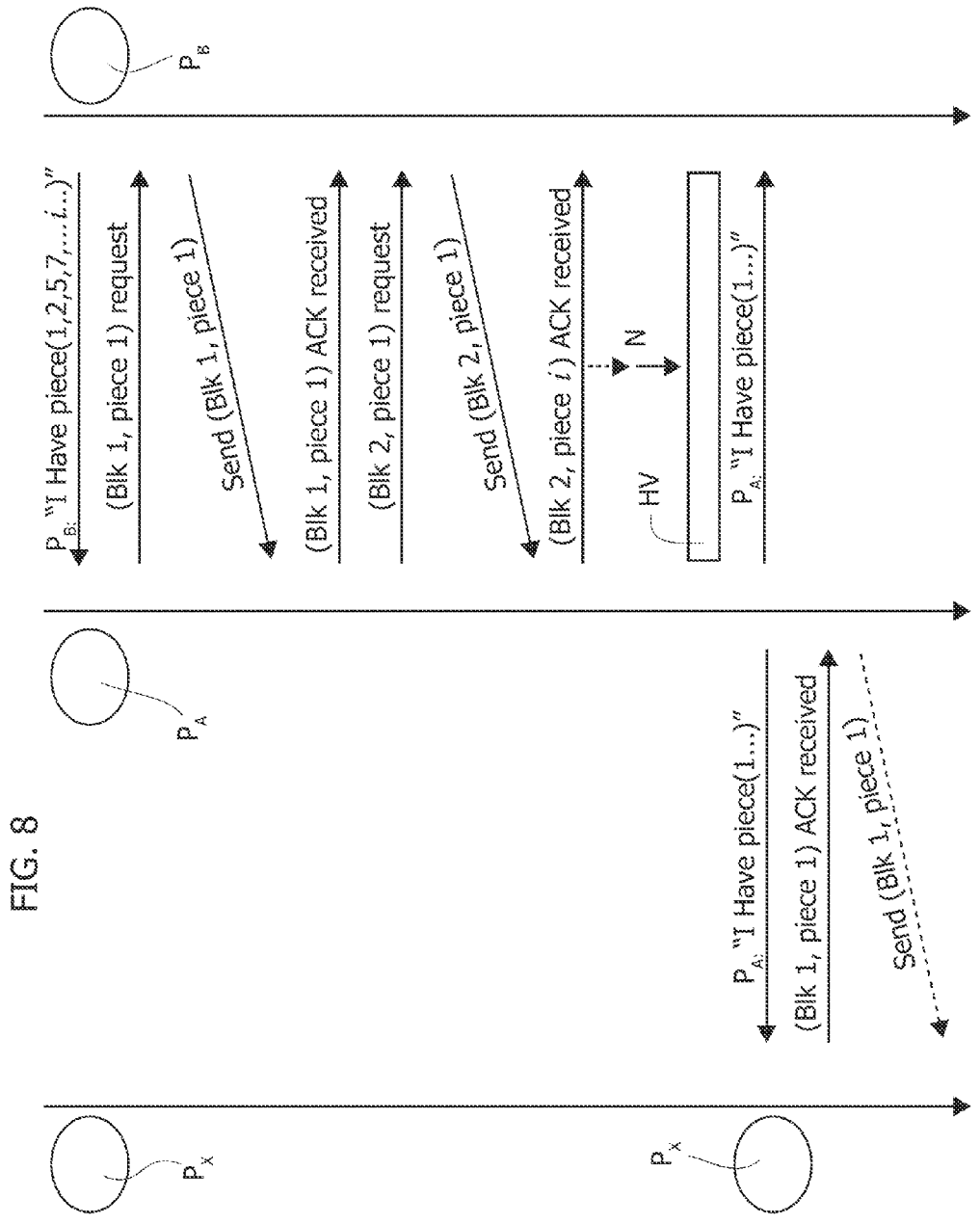
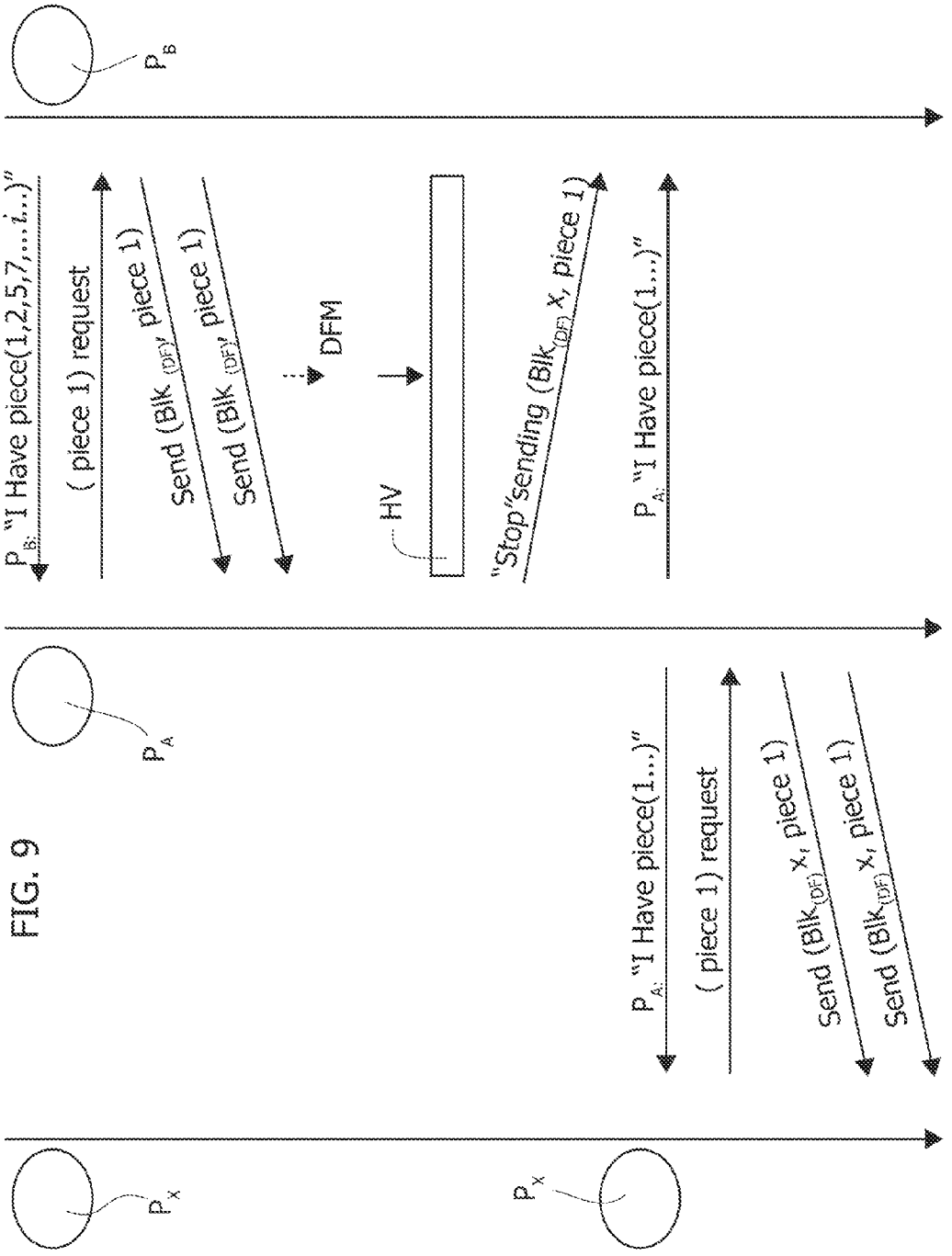

P_B
P_A
P_X
P_X
P_B: "I Have piece(1,2,5,7,...,i..)"
(Blk 1, piece 1) request
Send (Blk 1, piece 1)
(Blk 1, piece 1) ACK received
(Blk 2, piece 1) request
Send (Blk 2, piece 1)
(Blk 2, piece i) ACK received
N
HV
P_A: "I Have piece(1,..)"
P_A: "I Have piece(1,..)"
(Blk 1, piece 1) ACK received
Send (Blk 1, piece 1)

FIG. 8

FIG. 9


P_X
P_A
P_B
P_A: "I Have piece(1,,,)"
(piece 1) request
Send ($Blk_{(DF)}$, piece 1)
Send ($Blk_{(DF)}$, piece 1)
DFM
HV
"Stop"sending ($Blk_{(DF)}$ x, piece 1)
P_A: "I Have piece(1,,,)"
P_B: "I Have piece(1,2,5,7,...,i,..)"
(piece 1) request
Send ($Blk_{(DF)}$ x, piece 1)
Send ($Blk_{(DF)}$ x, piece 1)
P_X
P_X

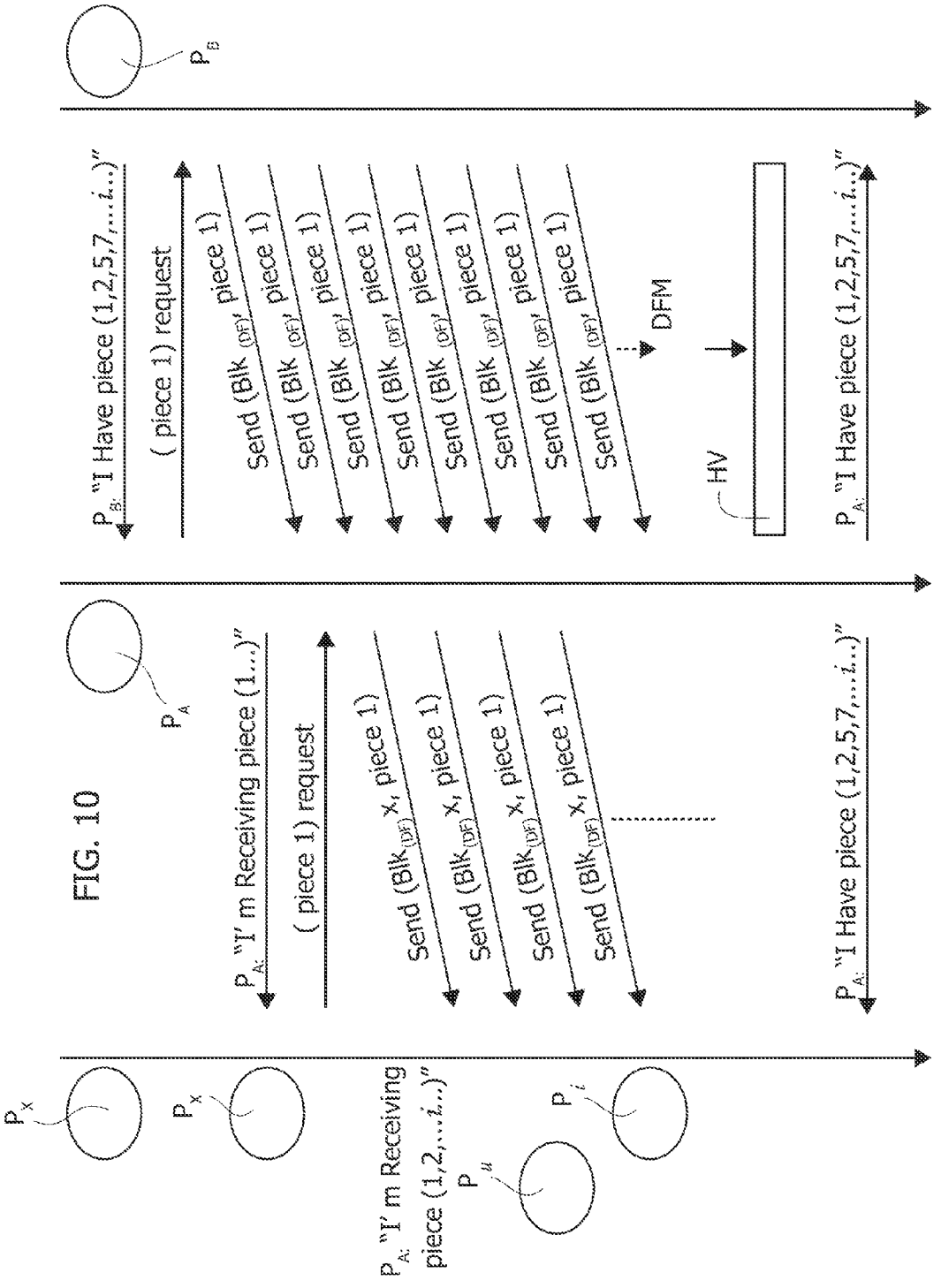

P_X
P_X
P_A: "I' m Receiving piece (1,2,...i..)"
P_u
P_i
P_A
P_A: "I' m Receiving piece (1...)"
(piece 1) request
Send (Blk_(DF) x, piece 1)
Send (Blk_(DF) x, piece 1)
Send (Blk_(DF) x, piece 1)
Send (Blk_(DF) x, piece 1)
P_A: "I Have piece (1,2,5,7,...i..)"
P_B
P_B: "I Have piece (1,2,5,7,...i..)"
(piece 1) request
Send (Blk_(DF), piece 1)
Send (Blk_(DF), piece 1)
Send (Blk_(DF), piece 1)
Send (Blk_(DF), piece 1)
Send (Blk_(DF), piece 1)
Send (Blk_(DF), piece 1)
Send (Blk_(DF), piece 1)
Send (Blk_(DF), piece 1)
DFM
HV
P_A: "I Have piece (1,2,5,7,...i..)"

FIG. 10

FIG. 11

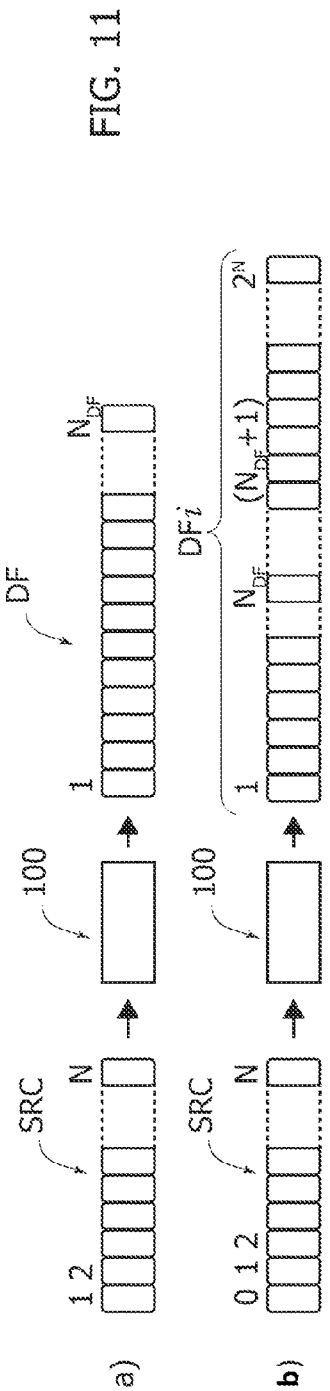


FIG. 12

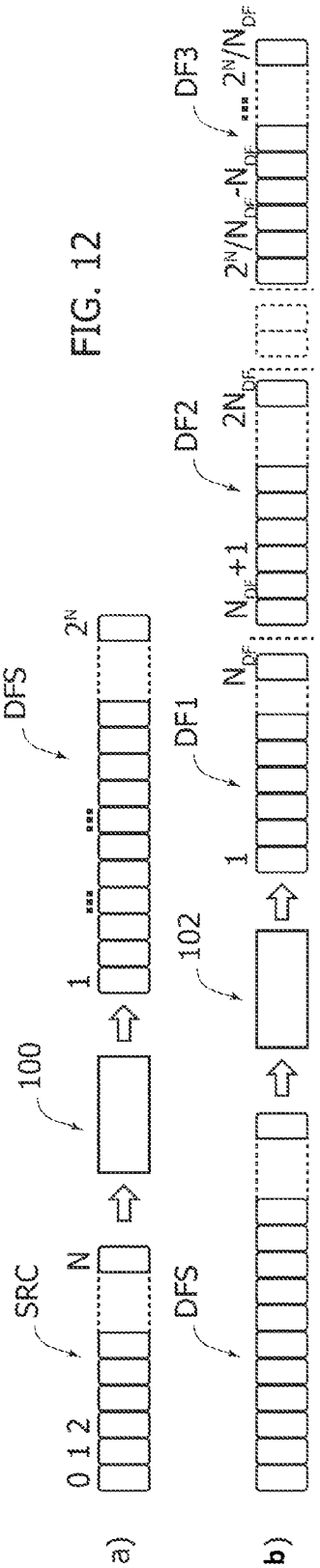


FIG. 13

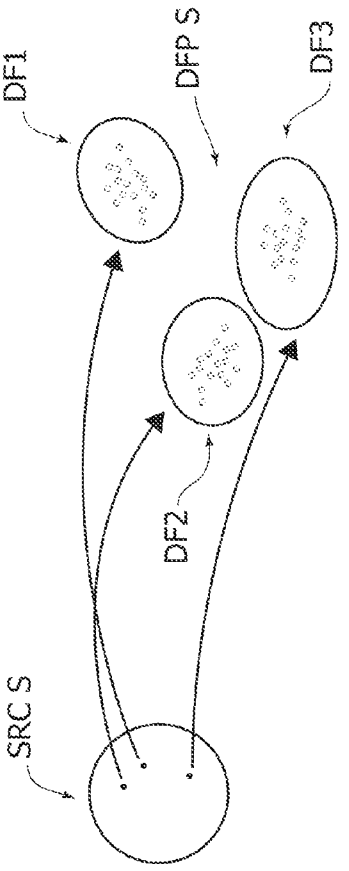


FIG. 14

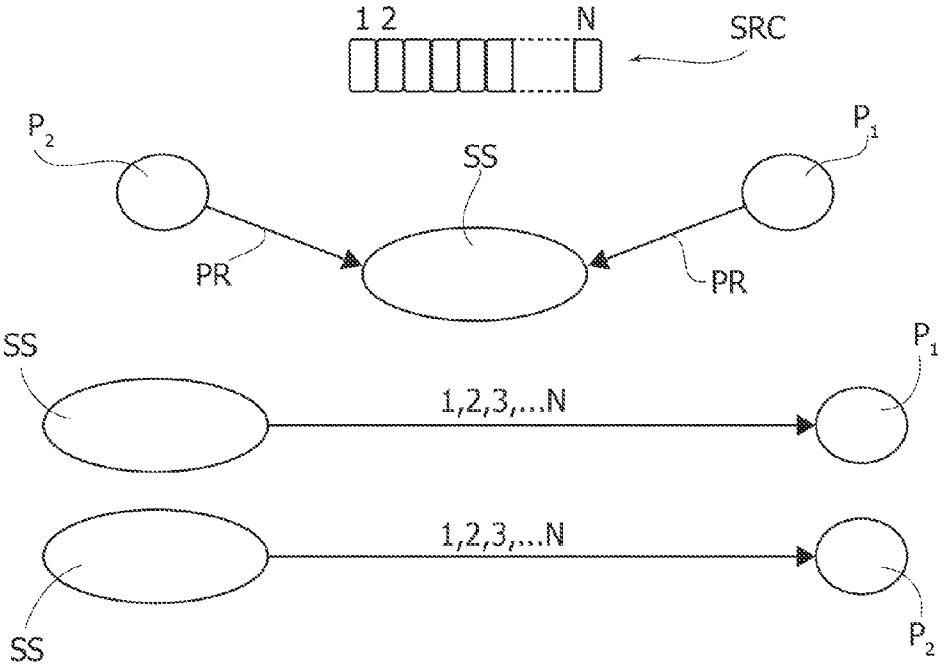

1 2
N
SRC
P2
P1
SS
PR
PR
SS
P1
1,2,3,...N
1,2,3,...N
SS
P2

FIG. 15

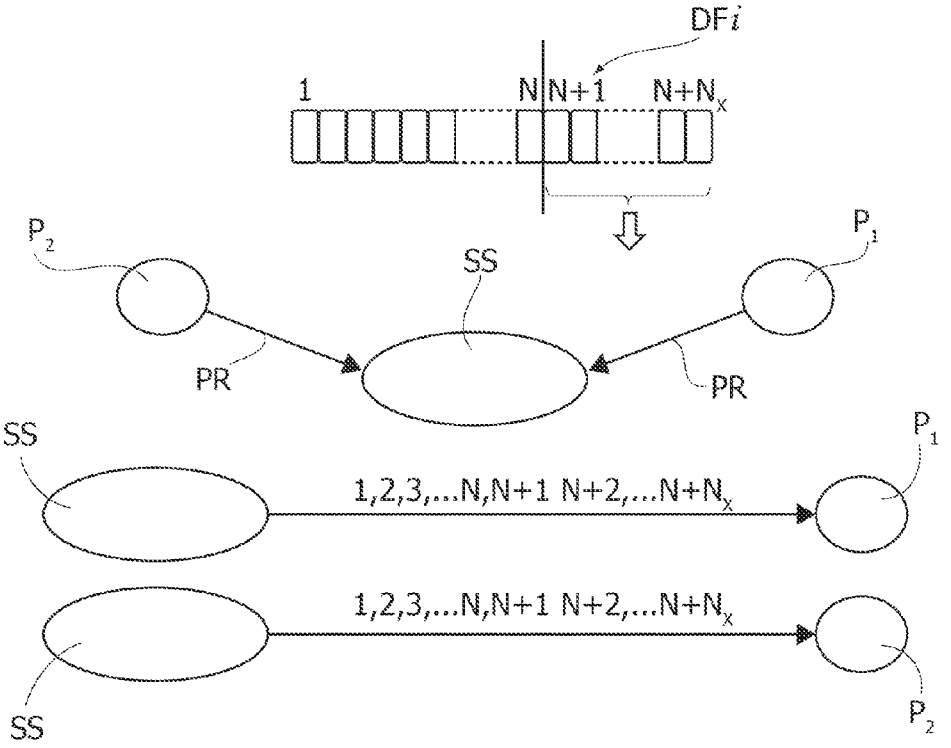

DFi
1
N
N+1
N+Nx
P2
P1
SS
PR
PR
SS
P1
1,2,3,...N,N+1 N+2,...N+Nx
1,2,3,...N,N+1 N+2,...N+Nx
SS
P2

FIG. 16

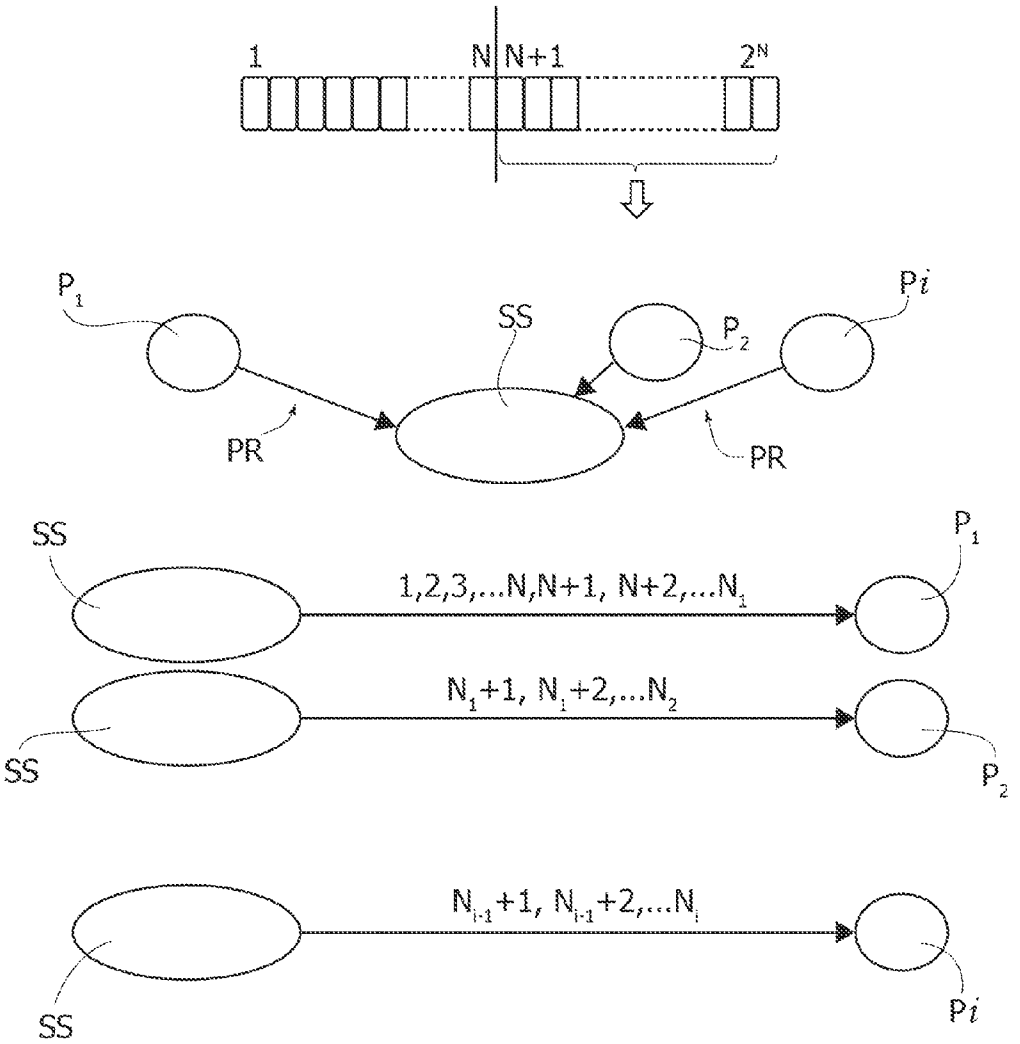

1
N
N+1
2^N
P1
SS
P2
Pi
PR
PR
SS
1,2,3,...N,N+1, N+2,...N1
P1
N1+1, N1+2,...N2
SS
P2
Ni-1+1, Ni-1+2,...Ni
SS
Pi

FIG. 17

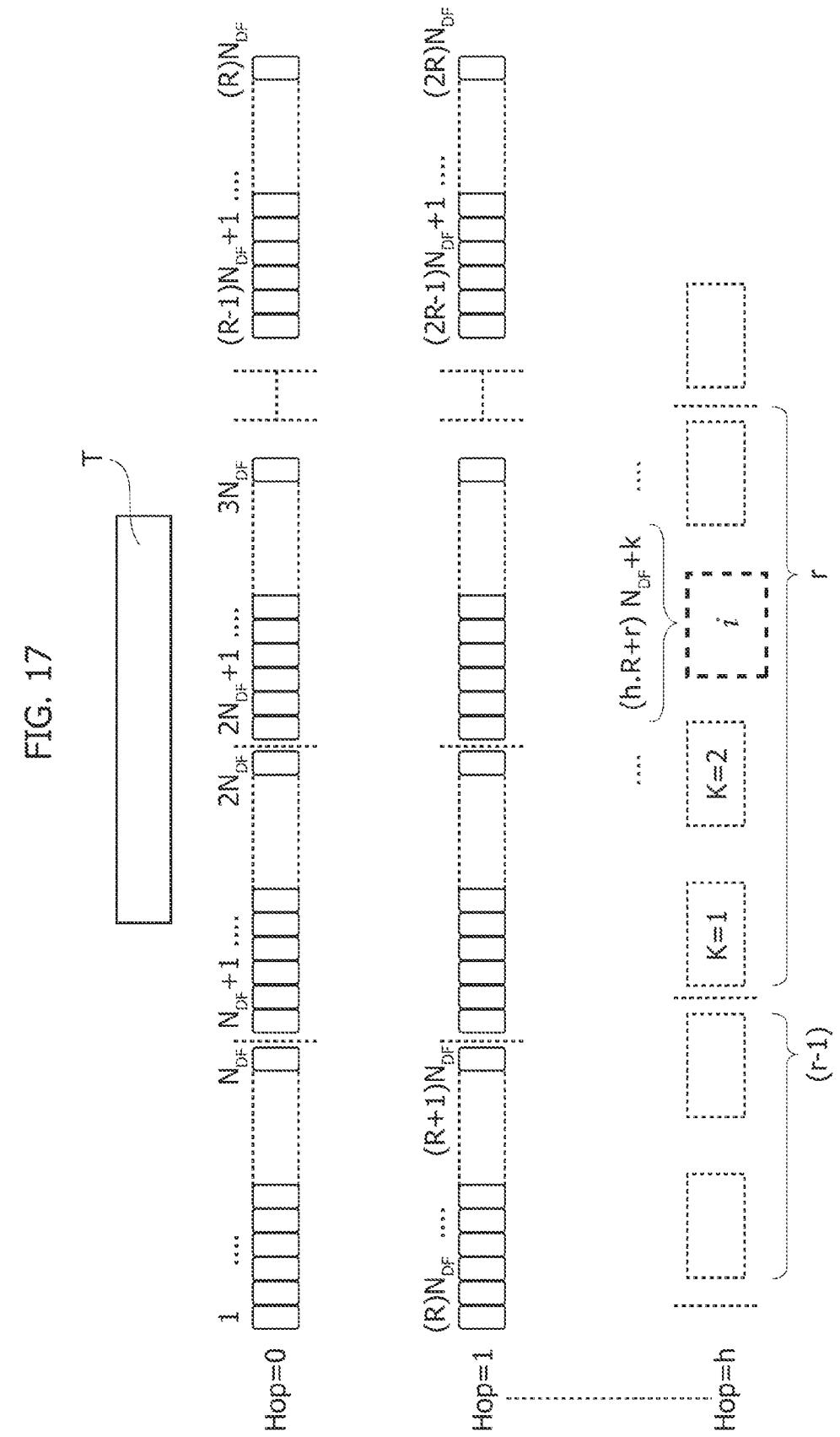

T
Hop=0
Hop=1
Hop=h
1 .... N_{DF}
N_{DF}+1 .... $2N_{DF}$
$2N_{DF}$+1 .... $3N_{DF}$
(R-1)N_{DF}+1 .... (R)N_{DF}
(R)N_{DF} .... (R+1)N_{DF}
(2R-1)N_{DF}+1 .... (2R)N_{DF}
.... (h.R+r) N_{DF}+k ....
K=1
K=2
i
(r-1)
r

FIG. 18

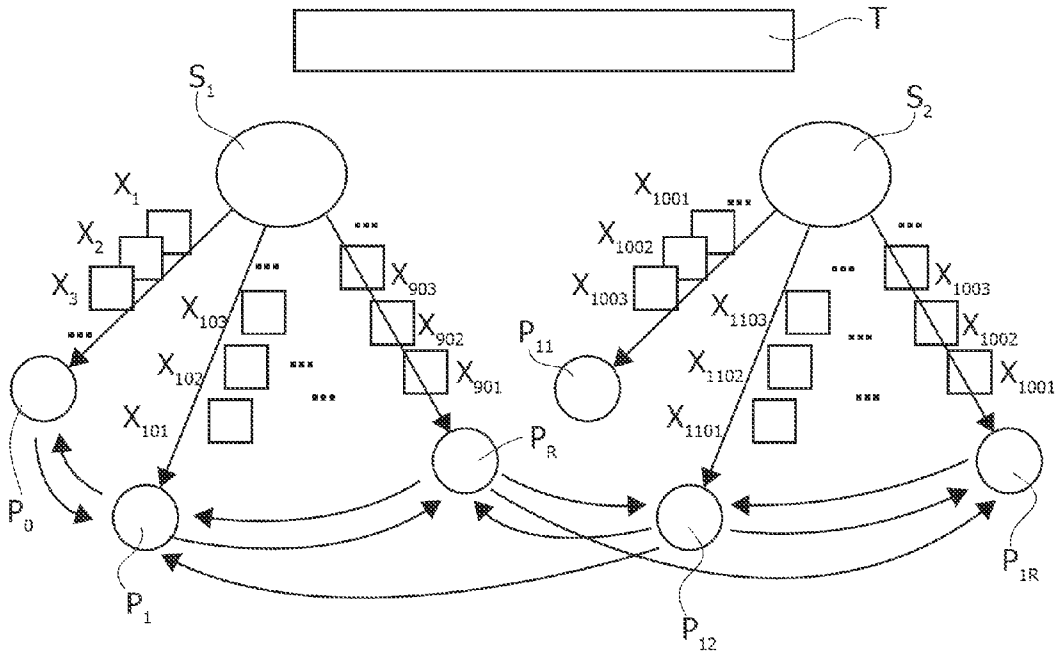

T
S1
S2
X1
X2
X3
X101
X102
X103
X901
X902
X903
X1001
X1002
X1003
X1101
X1102
X1103
P0
P1
P11
P12
PR
P1R

FIG. 19

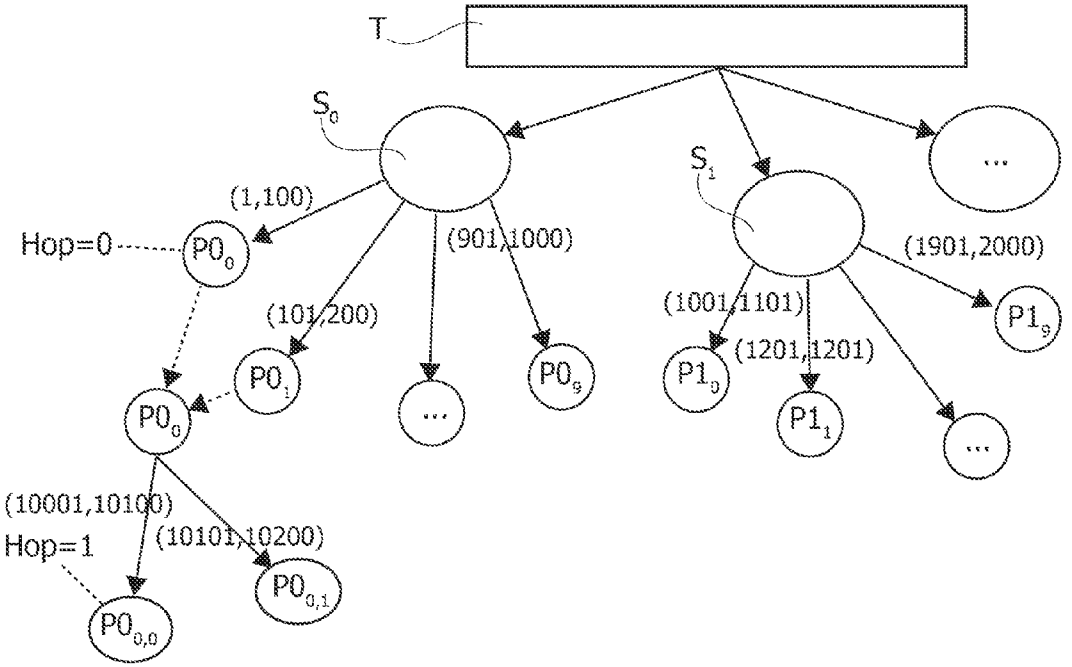

T
S0
S1
...
(1,100)
Hop=0
P0₀
(901,1000)
(101,200)
P0₁
P0₉
...
P0₀
(10001,10100)
Hop=1
(10101,10200)
P0₀,₀
P0₀,₁
(1901,2000)
(1001,1101)
(1201,1201)
P1₀
P1₁
P1₉
...

FIG. 20

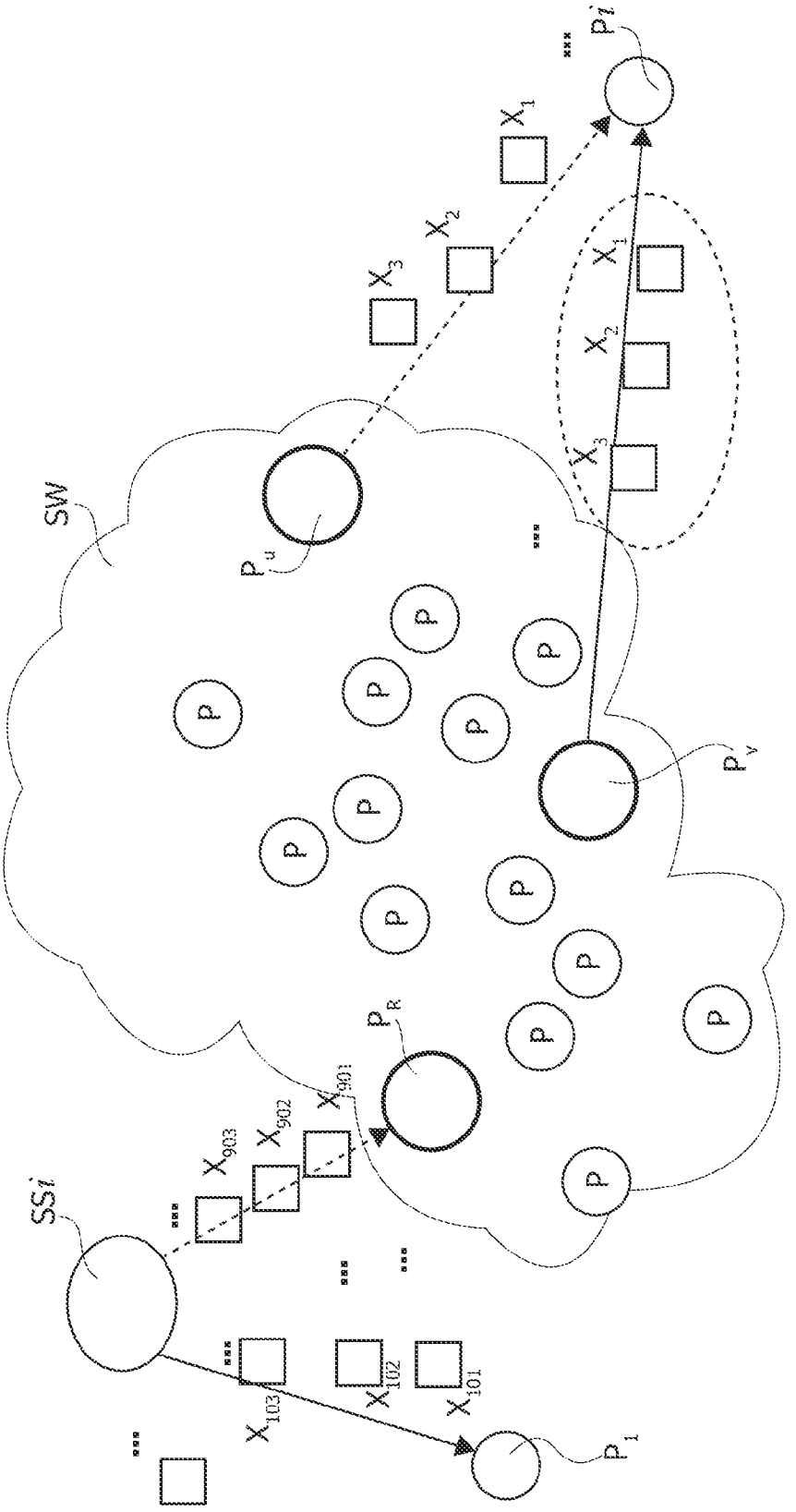


FIG. 21

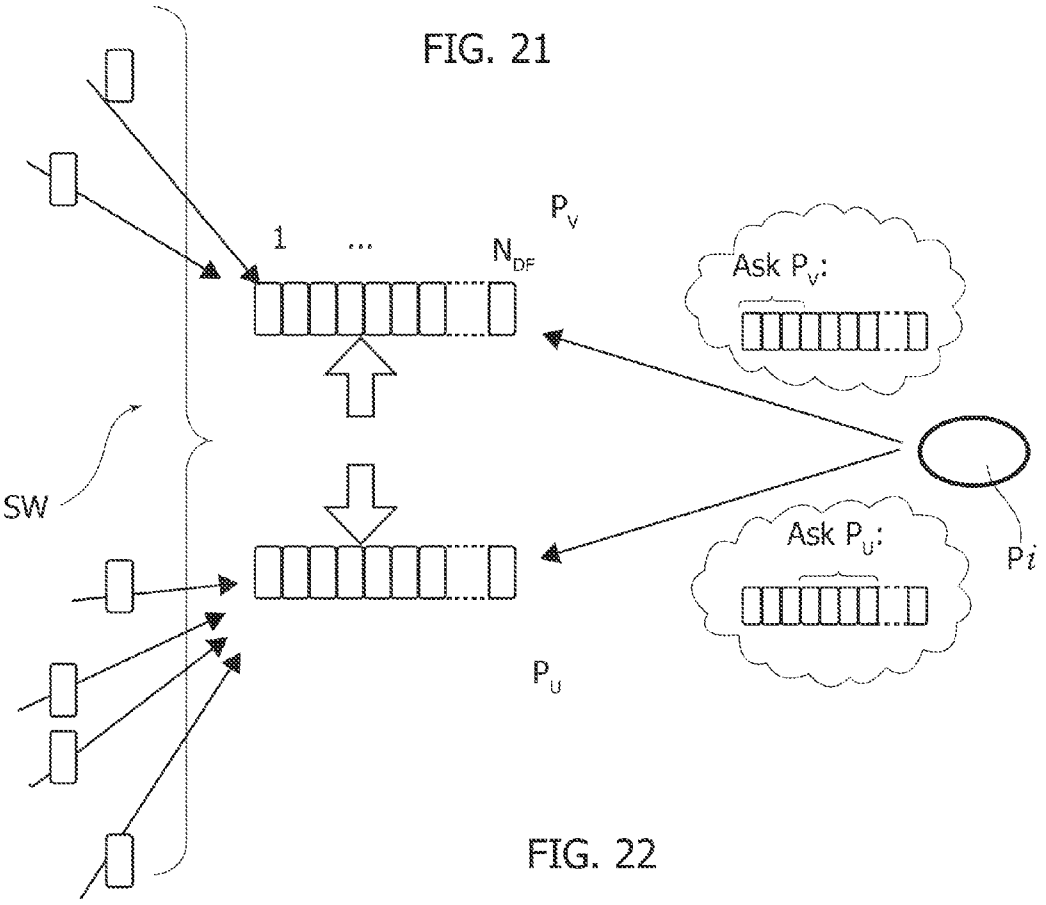


FIG. 22

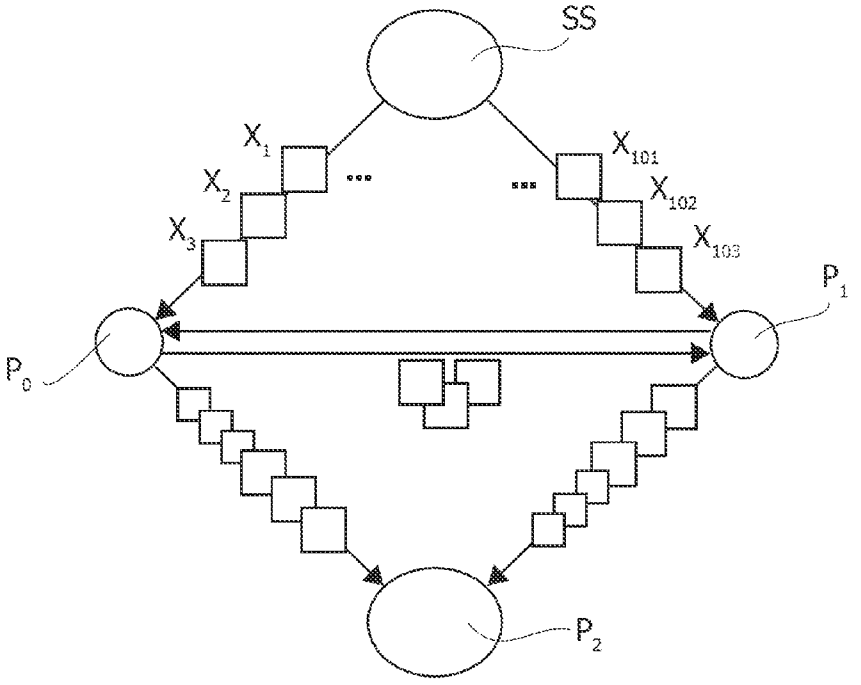


FIG. 23

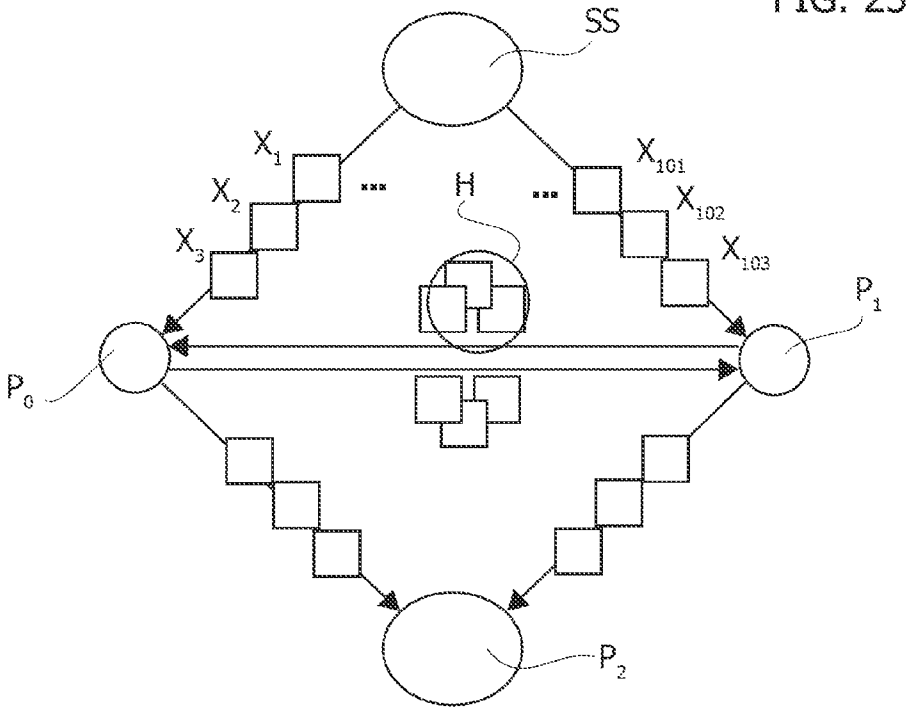

SS
X1
X2
X3
H
X101
X102
X103
P0
P1
P2

FIG. 24

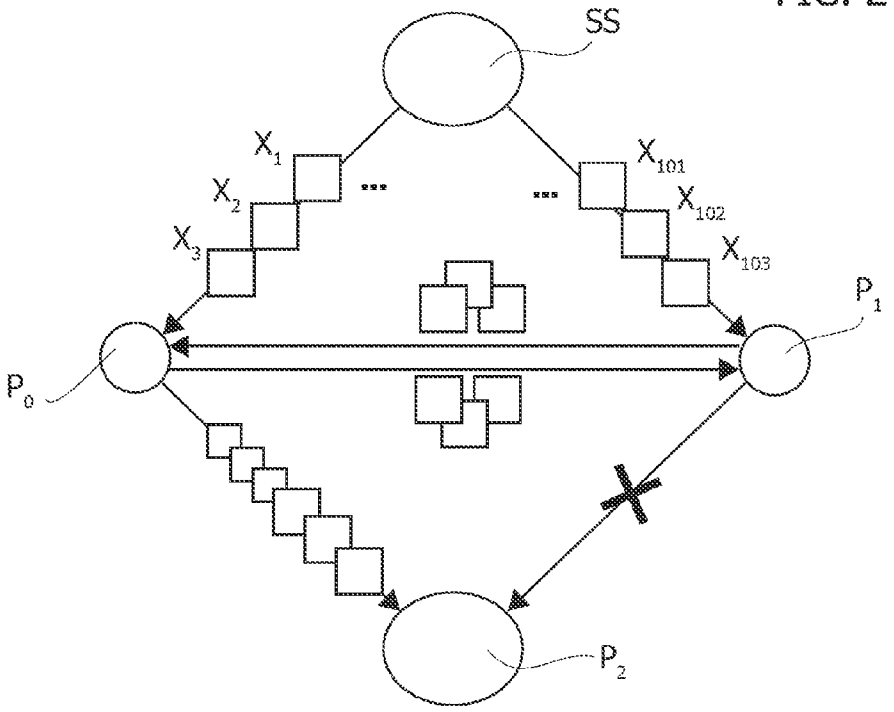
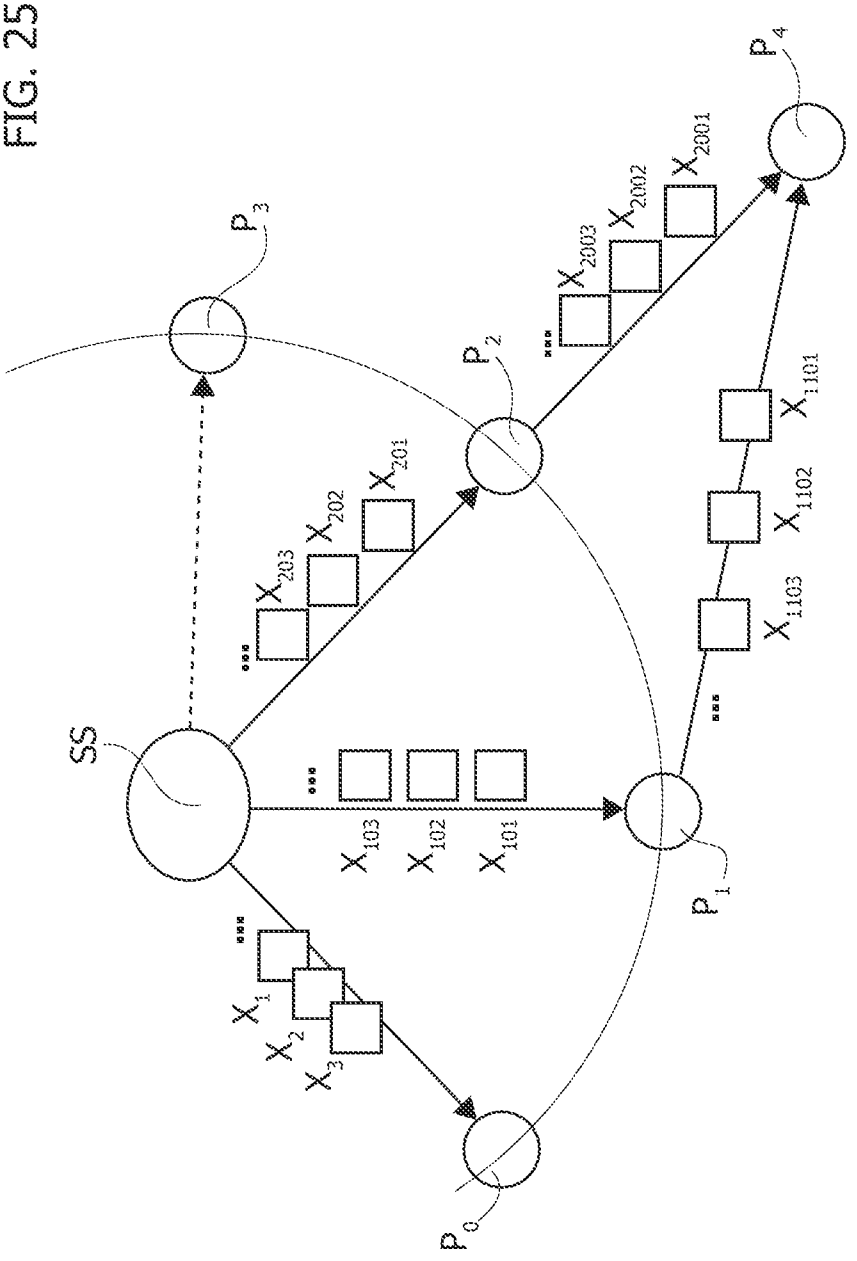

SS
X1
X2
X3
X101
X102
X103
P0
P1
P2

FIG. 25


SS
P0
P1
P2
P3
P4
X1
X2
X3
X101
X102
X103
X201
X202
X203
X2001
X2002
X2003
X1101
X1102
X1103

METHOD FOR DISTRIBUTING INFORMATION CONTENTS, CORRESPONDING DEVICE AND COMPUTER PROGRAM PRODUCT

PRIORITY CLAIM

[0001] The instant application claims priority to Italian Patent Application No. TO2013A000230, filed 22 Mar. 2013, which application is incorporated herein by reference in its entirety.

TECHNICAL FIELD

[0002] This disclosure relates to techniques for distributing information contents.

[0003] Various embodiments may apply to so-called cooperative networks such as, e.g., Peer-to-Peer networks for distributing media contents.

SUMMARY

[0004] Peer-to-Peer (P2P) networks are an environment created at the application layer by a local software application. The application can communicate with other users in the network running the same software and create an overlay network at the application layer where each end user may share his/her own contents and resources with the "peers" in the whole overlay.

[0005] The related Wikipedia™ entry available on the Internet at http://en.wikipedia.org/wiki/Network_coding, and articles such as:

[0006] A. Sentinelli et al.: "*A Survey on P2P Overlay Streaming Clients*", IOS press "Towards the Future Internet—A European Research Perspective," 2009, ISBN 978-1-60750-007-0 pp. 273-282;

[0007] G. Marfia et al.: "Digital Fountains+P2P for Future IPTV Platforms: A Testbed Evaluation," in Proc. 4th IEEE/IFIP International Conference on New Technologies, Mobility and Security, Paris, 2011, pp. 1-5,

[0008] which are incorporated by reference, provide technical background information in this area.

[0009] Commercial products such as the Octoshape™ content distribution network (CDN) (see, e.g., www.octoshape.com. and http://www.octoshape.com/?page=company/partners, which are incorporated by reference) propose a compromise between the bandwidth cost savings expected from P2P exploitation and reliability measured against 100% Quality of Service (QoS).

[0010] US 2011/0161457, which is incorporated by reference, discloses a network, such as a Peer-to-Peer overlay network, including a set of peer terminals wherein information codes are arranged in pieces comprised of chunks of bytes. A first peer identifies missing chunks in the received pieces and requests such missing chunks from other peers. The chunks are subjected to a fountain-code encoding wherein the chunks in a piece are X-ORed. The first peer is, therefore, capable of reconstructing a received piece encoded with fountain codes from a combination of linearly independent chunks corresponding to the piece. The chunks are transmitted over the network with a connectionless protocol, without retransmission of lost packets, for example, with a UDP. Embodiments of such an arrangement may combine the use of a Unit Data Protocol (UDP) for content delivery in cooperative networks (e.g., in a P2P file-sharing application that distributes block units encoded with a so-called Digital Fountain—DF) with the capability of supporting a system architecture with a higher degree of complexity in comparison to solutions involving single algorithms.

[0011] Various modules possibly included in such systems may benefit from further investigation and improvement. This may apply, for instance, to arrangements adapted to change the behavior of the peers in the network "on-the-field", that is during actual network operation, and/or to performance of a P2P protocol for content delivery when evaluated by using download time as a metric: the shorter the download time, the better the protocol.

[0012] Embodiments herein may improve the capabilities of conventional arrangements as discussed in the foregoing.

[0013] Various embodiments include a method, and may also relate to corresponding devices (e.g., network terminals) as well as to a computer program product, loadable into the memory of at least one computer (e.g., a terminal in a network), and comprising software-code portions adapted to perform the steps of the method when the product is run on at least one computer. As used herein, such a computer-program product is intended to be equivalent to a computer-readable medium containing instructions for the control of the computing system, so as to coordinate the execution of a method according to an embodiment. The reference to "at least one computer" is meant to highlight the possibility for an embodiment to be implemented in a modular and/or distributed form.

[0014] Various embodiments may provide a judicious balance of security constraints and reduced download time, with, e.g., hash-integrity checks delayed or possibly skipped. Various embodiments may thus provide the capability of speeding up the data-sharing mechanism by delaying data-integrity checks at the piece level, while also tackling the issue of possible redundancy of the blocks delivered in the network. Various embodiments may overcome performance bottlenecks associated with a constrained piece/block approach whereby, if a piece has not been fully downloaded and verified through, e.g., a hash-integrity check, it cannot be shared with other peers. In various embodiments, blocks may be shared before piece validation, which may lead to improvements in exploiting the upload bandwidth, while also dealing with the possibility that certain peers may distribute redundant blocks.

[0015] Various embodiments may speed up sharing of blocks by diversifying as much as possible the information distributed in the network. Various embodiments may provide a protocol for content delivery in cooperative networks (e.g., P2P networks), which protocol exploits information diversity among the blocks distributed in the network. Various embodiments may provide a P2P network system that maximizes, or approximately maximizes, information diversity among the blocks in the network by exploiting a Digital Fountain mechanism and ad-hoc changes in a protocol.

[0016] Various embodiments may be based on a development of the BitFountain (BF) approach of US 2011/0161457, which is incorporated by reference, which approach may be defined as a BitFountain Delivery Remap (BFDR) mechanism. Various embodiments may provide a P2P system based on the BFDR concept by means of a protocol for, e.g., streaming applications which enhances the diversity of the data blocks (e.g., Digital Fountain—DF blocks) shared in a network.

[0017] Various embodiments may provide alternative solutions to conventional BitTorrent (BT) or Torrent-like arrange-

ments. Various embodiments may provide improved performance over BT both for large piece size and for smaller (and more traditional) piece size.

[0018] Various embodiments may provide a Peer-to-Peer (P2P) system where a Digital Fountain (DF) and a UDP are used in, e.g., P2P file sharing at the application layer and the transport layer, respectively. Various embodiments may address issues related to, e.g., protocol design in terms of headers and encoding rules in data-block generation.

[0019] Various embodiments are suitable for use in a planetary environment including, e.g., hundreds of nodes. Various embodiments may cover scenarios with more than one seed supplier peer, including CDNs with servers that deliver the same content, while complying with inherent constraints and ensuring diversity in data generation.

BRIEF DESCRIPTION OF THE DRAWINGS

[0020] Various embodiments will now be described, by way of example only, with reference to the annexed figures, wherein:

[0021] FIG. **1** is exemplary of a file-piece-block information structure.

[0022] FIG. **2** is exemplary of XOR combinations in a Digital Fountain (DF).

[0023] FIG. **3** is exemplary of a coding matrix.

[0024] FIGS. **4** and **5** are exemplary of block sharing in a cooperative network.

[0025] FIGS. **6** and **7** represent block-distribution schemes in cooperative networks.

[0026] FIGS. **8**, **9**, and **10** are exemplary of block-delivery in cooperative networks.

[0027] FIGS. **11** and **12**, each including two portions designated a) and b), are exemplary of block generation in a digital fountain.

[0028] FIG. **13** exemplifies a digital-fountain-piece-space representation.

[0029] FIGS. **14**, **15**, and **16** exemplify various types of information diversity.

[0030] FIG. **17** is representative of an XOR tree for digital-fountain-block generation.

[0031] FIG. **18** is exemplary of use of seeds in embodiments.

[0032] FIG. **19** is exemplary of seeds and peers at multiple hops in embodiments.

[0033] FIG. **20** is exemplary of dealing with far peers in embodiments.

[0034] FIG. **21** schematically represents possible coordination of requests for blocks in embodiments.

[0035] FIGS. **22**, **23**, and **24** are exemplary of block forwarding in cooperative networks.

[0036] FIG. **25** is exemplary of bit-fountain-delivery remapping in embodiments.

DETAILED DESCRIPTION

[0037] The following table lists a number of acronyms that may be used throughout this description along with their meaning.

BF	Bit Fountain
BFDR	Bit Fountain Delivery Remap
DF	Digital Fountain
P2P	Peer To Peer
STB	Set Top Box
TCP	Transport Control Protocol
UDP	Unit Data Protocol

[0038] In the following description, numerous specific details are given to provide a thorough understanding of exemplary embodiments. The embodiments can be practiced without one or more of the specific details, or with other methods, components, materials, etc. In other instances, well-known structures, materials, or operations are not shown or described in detail to avoid obscuring aspects of the embodiments. Reference throughout this specification to "one embodiment" or "an embodiment" means that a particular feature, structure, or characteristic described in connection with the embodiment is included in at least one embodiment. Thus, the appearances of the phrases "in one embodiment" or "in an embodiment" in various places throughout this specification are not necessarily all referring to the same embodiment. Furthermore, the particular features, structures, or characteristics may be combined in any suitable manner in one or more embodiments.

[0039] The headings and references provided herein are for the convenience of the reader only and do not interpret the scope or meaning of the embodiments.

[0040] Current file-sharing applications may perform an integrity check after a full piece (or chunk) of a file has been downloaded; an established appreciation in P2P technology thus suggests that smaller pieces may lead to better performance.

[0041] It was noted that an element in that context may lie in data integrity being checked after a full piece has been downloaded. A basic assumption in BitTorrent policy dictates that it is not permitted to upload blocks, i.e., to share them with other peers, until the full piece has been downloaded and validated. In fact, blocks are the data units exchanged among the peers, and blocks represent the smaller unit components of pieces.

[0042] Various embodiments herein may be able to exchange Digital Fountain (DF) X-ORed blocks even if the piece is not yet fully downloaded. This by also noting that, if peers transmit block units as soon as they receive them, the improvement in resource exploitation due to immediate sharing may be at least partly canceled out by redundant information being undesirably produced in the network. This may possibly be the case even if blocks are shared through a pseudo-random combination of blocks that starts from one seed, which is the same for the whole P2P network, and proceeds through a "pseudo-random" extraction of the indexes that refer to the blocks which are XORed. It may be noted that the issue of redundancy is not addressed in US 2001/0161457, which is incorporated by reference.

[0043] In order to avoid uploading redundant information, various embodiments may implement a policy which aims at maximizing information diversity of the block combinations at each peer of the network without generating further overhead information.

[0044] In various embodiments, the download process may be speeded up by exploiting free upload resources from the very beginning of the download process. Diversity as offered by a Digital Fountain (DF) during the generation of data blocks may be exploited in order to avoid that early sharing may produce redundancy among blocks (i.e., that one peer

might receive the very same blocks from two different suppliers), while changes in the protocol headers may avoid redundancy loops.

[0045] Various embodiments may be based on remapping Digital Fountain (DF) block generation for the various peers in order to maximize the information diversity in the whole network swarm.

[0046] FIG. **1** is exemplary of a file-piece-block information structure as used in cooperative networks (e.g., P2P) both in BT and BF (e.g., US 2011/0161457, which is incorporated by reference).

[0047] In such embodiments, the content to be delivered over the network is organized as files F, pieces PE (sometimes referred to also as "chunks") and blocks B.

[0048] A file F is made of pieces (e.g., PE_0, PE_1, PE_2, . . .) and a piece PE_i is made of blocks B. The block B is the atom unit for content delivery over a cooperative network (e.g., P2P).

[0049] A basic rationale of operation of such a network may be as follows:

[0050] I (a node in the network) am missing a piece of the file→I ask for that piece

[0051] I ask for a piece→I receive blocks of that piece.

[0052] I am asked for one piece→I send blocks of that piece.

[0053] A P2P network may include peers, seeds, and a tracker.

[0054] A seed is a terminal or "node" in the network that holds the whole file. A seed may be assumed only to supply contents, without ever asking for any contents for downloading.

[0055] A peer is a node or client in the network that has some piece (or chunk) of the file but not the whole file. A peer may hold some pieces that have been validated, e.g., via a hash integrity check, and those pieces can be shared over the network with other peers: the (first) peer acts as a supplier to the other peers because it supplies information; in conventional BT, such a peer cannot share pieces that have been (only) partially downloaded.

[0056] A tracker is an entity, usually hosted by a reachable server, that updates the list of peers and seeds with respect to a certain file.

[0057] A process considered herein may include a check such as a hash check (or piece signature, or data integrity check). In fact, malicious peers may manipulate the content of a block; a mechanism may thus exist at the piece level in order to validate data integrity and prevent error propagation towards the rest of the peer population.

[0058] Joining a P2P "swarm" may involve downloading a "Torrent" file (a few Kb) containing all the hash codes (or the signatures) of the pieces in that file. Therefore, when reception of the blocks of a specific piece PE_i has been completed, a hash may be computed and compared with an "unofficial" hash corresponding to PE_i, which is written in the Torrent file.

[0059] Comparison between the two hash codes may yield two outcomes:

[0060] the hash codes are not equal: integrity of the data in that piece is negated→the piece is discarded;

[0061] the hash codes are equal: data integrity of that piece is confirmed and the piece can be shared with the rest of the network.

[0062] In such an arrangement, the piece and/or the blocks of the piece can be shared (=uploaded) only after piece validation.

[0063] The Digital Fountain (DF) mechanism of, e.g., US 2011/0161457, which is incorporated by reference, may involve an XOR sum of blocks: one piece is made by a fixed number of blocks B (configurable by the user) that may be encoded through the DF. FIG. **2** shows how new blocks of distributed information, denoted X_1, X_2, X_3, . . . can be produced from the blocks B of an original piece PE_i through simple XOR combinations. In Bit Fountain (BF), a piece may be "decodable" (i.e., "reconstructable", that is adapted to be reconstructed) only when a sufficient number of (linearly independent) encoded blocks have been received: from a set of N blocks there may be 2^N different XOR combinations but only N independent blocks may be necessary to decode one block.

[0064] An exemplary coding matrix DFM for a file made of three pieces may be as shown in FIG. **3**; such a matrix mixes three blocks from a file through XOR operations to produce a new DF block over which the information is distributed. When that block (shown on the left) becomes sufficiently "large" (i.e., "at least N independent blocks"), the matrix can decode the information, i.e., reconstruct the piece from the corresponding set of blocks B.

[0065] In BT, as in a BF implementation, a piece may be reconstructed, and thus shared, only after the hash integrity check. Such an approach is schematically represented in FIG. **4**, where a node P_0 is shown receiving blocks from a seed supplier SS, and only after receiving a last block LB and performing successful validation, e.g., Hash Verification HV of the (fully downloaded) piece, the node P_0 starts sending the blocks to another node P_1.

[0066] FIG. **5** is exemplary of an approach where a node P_0 receiving blocks from a seed supplier SS starts sending the blocks to another node P_1 without waiting for data-integrity validation, that is, before the piece becomes "reconstructable" by fully downloading the set of blocks received from the seed supplier SS and corresponding to the piece.

[0067] Various embodiments of such an approach may thus involve relaxing the security checks in order to improve the overall performance of the system by better exploiting the overall upload resources, making upload faster by reducing the average download time at the peer.

[0068] Various embodiments may thus involve a change in the local protocol so that blocks start to be shared before performing the data-integrity check (e.g., hash check).

[0069] In various embodiments, this may lead to the integrity check being delayed, while not necessarily dispensed with, i.e., "skipped": that is, blocks start to be sent from P_0 to P_1 without waiting for data-integrity validation. By adopting the approach of FIG. **5**, the peer P_0 may share data to P_1 as soon as a block is received.

[0070] In various embodiments, a check may be maintained in order to avoid error propagation towards the whole P2P swarm: in such a case, an error may not be detected during the sharing process and, therefore, may be recognized only by the end user, so to say "manually", that is when the content is opened and played, after full-content download.

[0071] A data-integrity check may represent a basic verification test in those P2P environments where single users may have malicious intentions, that is, where data can be manipulated. This simply represents a matter of network trust: the more one node may trust its neighbors (and, also, the network conditions), the more it can relax security measures, with network trust going with the need of security in P2P scenarios. In an environment of Set Top Boxes sharing content in

a network made of gateways, routers, and peers distributed over the whole world, security may require, at the application layer, specific checks, tests, and verifications that are written into the P2P protocol, a data-integrity check possibly being just one of these checks, tests, and verifications.

[0072] This means that protocol arrangements may be somehow "tuned" to speed up the overall download time, by observing that, while one may not trust 100% a P2P network, a judicious compromise may be arrived at between speed and security. In the past, security has been a basic point in P2P networks because software could be downloaded for free and any participant in the system could pollute the P2P network by resorting to very simple "tricks". Requirements in terms of security may, however, be relaxed by noting, e.g., that a STB can host a private encrypted network that is close to end users.

[0073] Redundancy may be generated by the immediate sharing of identical blocks among peers that are somehow close to the same supplier. Especially at the beginning of the download session, if only one seed is available, each peer can forward what it has just received. Identical blocks may thus diffuse over the network and be shared among the peers. Especially in large P2P networks, a peer which is far from the root may ask for new pieces from two suppliers and receive exactly the same blocks from both of them: bandwidth may be saturated and the global-network efficiency consequently reduced.

[0074] FIG. **6** is exemplary of a situation that may arise if the issue of information redundancy during the file-sharing process is not addressed: the two nodes P_1 and P_2 share the same information received from a seed supplier SS, and the node P_4 may receive from both P_1 and P_2 the same information, with undue redundancy.

[0075] FIG. **7** is exemplary of a mechanism which, in various embodiments, may enhance the diversity of information that flows into the network: thanks to differentiation in the content delivery (greater information diversity), redundancy among blocks that flow in the network is reduced.

[0076] In various exemplary embodiments of the Bit Fountain Delivery Remap (BFDR) arrangement considered herein, the P2P network may rely on peers that share blocks as soon as they are received: in various embodiments, in order to deal with the redundancy issues discussed previously, the possibility is considered for the protocol to link digital-fountain or DF block delivery and digital-fountain block generation with the target of promoting diversity of the information distributed into the network.

[0077] In various embodiments, a goal pursued may be sharing, as much as possible, new information to generate models to coordinate and diversify distribution, by, e.g., having DF blocks that are (first) all different and thereby creating a model to generate different DF blocks, i.e., a model adapted to diversify as much as possible the information shared in the network.

[0078] Various embodiments may thus provide a file-sharing scenario, e.g., for video-streaming distribution in a P2P network, wherein the virtues of the DF mechanism are exploited to diversify as much as possible the DF generation.

[0079] In various embodiments, this may involve a development of the Bit Fountain (BF) protocol disclosed in US 2011/0161457, which is incorporated by reference. Various embodiments may follow the main rules of the sharing mechanism just as is the case in Bit Torrent, while making download faster by delaying the validity (integrity) check of the information for the pieces.

[0080] In various embodiments, the risk that data integrity might be put at stake by malicious peers may be taken into account by noting, e.g., that:

[0081] a network made up of Set Top Boxes (STBs) may by itself be closed and secure: the peers, in fact, require an authentication;

[0082] other techniques, such as those known as "byzantine generals", where the peer asks for good references from neighbors, raise the level of security and prevent content sharing with bad peers.

[0083] Various embodiments may also address the issue of redundancy through a sharing mechanism which aims at maximizing the diversity of information shared in the network through different DF blocks.

[0084] Various embodiments may involve one or more of the following features:

[0085] validation, e.g., hash validation, may be delayed since (from the protocol's point of view) blocks are shared before piece validation. In real terms, the hash may not be necessarily "displaced", while blocks being shared before piece validation causes the protocol to behave differently: the peer shares blocks while not yet having reconstructed the piece and computed the hash, that is, with validation "delayed";

[0086] DF block generation may be based on the recognition that it is possible to build different DF blocks from the same blocks, and that such diversity may be exploited to avoid redundancy;

[0087] delivery of DF blocks may be "remapped" according to the peers in the network;

[0088] possible redundancy delivery loops may be dealt with according to various options.

[0089] FIGS. **8**, **9**, and **10** are illustrative of various block-delivery strategies in file-sharing systems, as adapted to be implemented in cooperative networks (e.g., P2P):

[0090] conventional BT protocol,

[0091] BF protocol (i.e., US2011/0161457, which is incorporated by reference),

[0092] various embodiments herein.

[0093] The sequence diagrams of FIGS. **8**, **9**, and **10** thus refer to arrangements where information content arranged in pieces including blocks of bits is sent over a network including a first peer terminal (i.e., P_A) and at least one second peer terminal (i.e., P_X). In such arrangements, a set of blocks B of a corresponding piece of information PE are received at the first peer terminal P_A (e.g., from a "seed" P_B, which may be another peer terminal). The corresponding piece is reconstructed from the set of blocks B received at the first peer terminal P_A.

[0094] The sequence diagrams of FIGS. **8**, **9**, and **10** refer to a situation where a new piece i has been downloaded by a generic peer; a protocol is set up to advise all the adjacent peers through a so-called "Have" message saying e.g.,: "I Have Piece 1, 2, . . . i . . . ".

[0095] Such a message lets the neighbors know "who has what", thus enabling them to forward to a peer who has a certain piece requests for such a piece they "want" to download.

[0096] In the arrangement exemplified in FIG. **8**, blocks are shared among peers only after a piece has been validated. For instance, the peer P_A may receive from P_B the message that some piece is available for download. P_A then may ask P_B for Piece 1. P_B answers by sending the corresponding blocks.

[0097] The logic model as portrayed in FIG. **8** is otherwise conventional, thus making it unnecessary to provide a more detailed description. When P_A has received from P_B all the blocks that make it possible to reconstruct Piece 1, then P_A executes piece validation through, e.g., a hash-integrity check HV of the reconstructed piece. The peer P_A will send a Have message to update its neighbors. P_X will receive the message and forward to P_A a request for Piece 1. P_A will be able to share the new content of Piece 1 with another peer P_X only after piece validation.

[0098] In the arrangement exemplified in FIG. **9**, i.e., Bit Fountain or BF (see, e.g., US2011/0161457, which is incorporated by reference), the pieces of information PE distributed over the network are fountain encoded by XOR-ing the blocks in a piece so that a received piece may be reconstructed (i.e., is "reconstructable," e.g., via the matrix DFM) from a combination of a corresponding set of linearly independent XOR-ed blocks (such as X_1, X_2, X_3, . . . in FIG. **2**).

[0099] In that case, information delivery is not constrained by any specific block request, in that no request for a specific block is made. Information is "spread" or (intuitively) "interleaved" over many blocks (X_i) via the digital-fountain mechanism. Thanks to the digital-fountain encoding mechanism, when P_A receives a sufficient number of blocks, P_A will be able to invert the DF matrix (as indicated at DFM in FIG. **9**) and reconstruct Piece 1, which becomes reconstructable when the DF matrix (see, e.g., FIG. **3**) becomes invertible. Then P_A will execute piece validation (e.g., a hash integrity check) HV just as shown for BT in FIG. **8**.

[0100] Again, only after piece validation, P_A will be allowed to share the new content of Piece 1 with another peer P_X. In the arrangement of FIG. **9**, delivery is no longer linked to a specific block request. The peer P_X sends all at the same time a group of requests for blocks related to piece 1. Any DF block is valuable in order to collect information so that, rather than collecting the "right" blocks related to the piece, the steps of inverting the DF matrix—in other words having enough (linearly independent) blocks to invert the matrix in order to reconstruct the piece—and validating the piece are at the basis of piece transfer. Once validation succeeds a "stop" message may be sent from P_X to P_A to cancel all pending requests related to the same Piece 1. Also, no acknowledgements are needed to notify that the block has been properly received: P_X sends a group of block requests to P_A all at the same time, and when the DF matrix can be inverted, P_X sends only one message to P_A to delete all remaining pending requests.

[0101] Through the integration of the digital fountain (DF) mechanism of, e.g., US2011/0161457, which is incorporated by reference, various embodiments may be somewhat moving from a "pull" model, where P_B answers to the needs of P_A, to a "push" model, where P_B sends blocks without signaling overhead. In order to fully exploit such a digital-fountain mechanism, various embodiments may "break" the validation constraint inherent to the fact that the piece validation step may prevent P_A from sharing the partial download of Piece 1 to other P_X's.

[0102] In various embodiments as schematically illustrated in FIG. **10**, the delivery of the partial download of the Piece 1 may be authorized as soon as P_X has notified its neighbors that it has no block data relating to Piece 1 and that P_A, which is a neighbor to P_X, has notified neighbors that it "is receiving" (e.g., from P_B) blocks relating to piece 1.

[0103] Stated otherwise, as soon as P_A has downloaded a block from P_B, it can forward that block to another peer, e.g., P_X, which means that the first peer terminal (P_A) starts sending to the (at least one) second peer terminal P_X blocks from the set of blocks being received (from P_B) before the corresponding piece of information becomes reconstructable from the set of blocks received.

[0104] To that effect, the relative protocol may be supplemented to include a new message for sending by, e.g., P_A, namely "I am Receiving piece (1, 2, . . . i . . .)".

[0105] In case of a long-distance connection, the presence of a small-signaling control message may cause appreciable differences not only as regards the time of download of the whole swarm of peers, but also in the management of the messages (and in complexity for developers).

[0106] Various embodiments may thus take advantage of the possibility of making assumptions, e.g., in terms of security in a (semi) controlled network such as a STB network.

[0107] Various exemplary embodiments may take advantage of the fact that the number of possible DF blocks associated with a same source piece may be very high and may be exploited to deal with the issue of redundancy.

[0108] FIGS. **11** and **12**, each including two portions designated a) and b), are representative of how DF blocks may be generated with respect to pieces in a conventional BT protocol, in BF (i.e., US2011/0161457, which is incorporated by reference), and in various embodiments herein.

[0109] For instance, a conventional BF approach—FIG. **11** a)—provides for each original source piece SRC, including N blocks, being converted via a DF procedure **100** into a single "DF piece" including a number N_{DF} of DF blocks.

[0110] In fact the same original piece SRC may be represented by many different DF pieces, generated by different XOR combinations among the N blocks of the original piece SRC. FIG. **11** b) schematically shows how the DF procedure **100** may generate a plurality of up to 2^N different DF blocks DF_i with i=1, . . . , 2^N, which is the maximum number of possible combinations of N initial blocks in the piece SRC.

[0111] As schematically represented in FIG. **12** a), such a plurality of up to 2^N different DF blocks may be seen as a DF sequence DFS which, as schematically represented in FIG. **12** b), may be split at **102** into DF sections DF1, DF2, . . . where each such DF section contains the whole information of one original piece SRC.

[0112] The correspondence between the original source space and the DF piece/section space may be represented as shown in FIG. **13**, that is as a correspondence between a source-piece-space domain SRC S and a DF piece/section space DFP S. Each piece in the domain of the source space SRC S corresponds in the DF piece/section space co-domain DFP S to a DF sequence DFS which is a set of DF pieces/sections, where each piece/section is made of N_{DF} DF blocks.

[0113] Although "DF piece" and "DF section" are conceptually equivalent and represent the same type and amount of information in the DF piece/section space, the rest of this description will refer to the "DF sections" in order to highlight the possibility of identifying a DF piece through an index in the DF sequence DFS DF1, DF2, DFi, . . .).

[0114] In brief:

[0115] a source piece SRC is just a piece of the original file F, which may be split into pieces or portion of files: the information therein has not been mixed, interleaved, manipulated; it is in the "pure" form it had in the original file;

[0116] a source piece block is a block of the original file: a piece is in fact comprised of blocks. A source piece SRC is made up of its related source blocks 0 to N. Again, the information has not been mixed, interleaved, manipulated; it is in the "pure" form it had in the original file;

[0117] N denotes the number of blocks in one piece SRC from the original file;

[0118] each DF block is generated by XORing a set of source blocks;

[0119] for each source piece SRC it is possible to build 2^N DF blocks. The sequence of DF blocks generated by the DF procedure **100** until the whole possible set of XOR combinations has been covered, is referred to as a DF sequence DFS. The sequence DFS is associated with only one original piece SRC;

[0120] a DF section (DFi) is a portion from the file after DF processing **100**; each DF section is comprised of N_{DF} blocks, i.e., N_{DF} is the number of DF blocks in one section; the whole DF sequence DFS may be split into DF sections DF1, DF2, . . . ; each DF section contains all the information of the original piece SRC insofar as each DF section represents a different XOR combination of blocks of the original piece.

[0121] For instance, an (original, i.e., source) piece SRC made of 100 blocks may be considered (of course as a merely exemplary case, not binding for the embodiments). If N=100 and N_{DF}=1024 (i.e. a highly redundant DF block) there are (still) $2^{100}/2^{10}=2^{90}$ possible DF sections related to the same original piece SRC.

[0122] In fact 2^{90} DF sections being possible for the same initial piece SRC is a very high number.

[0123] Consequently, in various embodiments, each piece request can be associated with a different DF section, and still share information of the same piece SRC.

[0124] A FIG. **14** is exemplary of a conventional BT arrangement where a peer P_1 asks (i.e., places a piece request PR) for a piece to a node acting as the "supplier" SS. Then P_2 peer asks the supplier SS for the same piece (i.e., same piece request PR) and the supplier SS, again, sends exactly the same blocks to both P_1 and P_2. As the blocks 1, 2, 3, . . . , N (see bottom part of FIG. **14**) are just "replicated" in the network for both P_1 and P_2, no information diversity is present.

[0125] In a Bit Fountain (BF) arrangement as exemplified in FIG. **15**, the pieces shared in the network are all made of DF blocks. Each DF piece/section may have, e.g., N=100 DF blocks: those blocks represent the whole information needed to recover the original information for reconstructing the source piece SRC. Since sudden network congestions or bandwidth oscillations may disrupt the flow of blocks over the network, the peers may miss some blocks among the N blocks sent by the supplier to the peer. As each DF block can be useful to invert the DF matrix and reconstruct the original piece, the supplier SS just keeps generating and sending more DF blocks: for example N_x as indicated in FIG. **15**, which may be equal to, e.g., 50. So, whenever P_1 asks for a piece (piece request PR), the supplier keeps sending blocks to P_1 until a number of $N+N_x$ (150, in the non-binding example made here) blocks is reached. If another peer P_2 asks for the same piece (same piece request PR) the supplier SS sends the same $N+N_x$ (e.g., 150) DF blocks. As schematically shown in FIG. **15**, in terms of information diversity, the network is hosting 50 new DF blocks per piece.

[0126] In various embodiments as schematically represented in FIG. **16**, further information diversity may be pursued by further exploiting the DF mechanism. By referring again, by way of non-limiting example, to a piece including N=100 blocks, it is possible to generate a high number of DF blocks (2^{100} possible DF blocks). As discussed in the foregoing, the whole DF sequence DFS can be split into a high number of sections (in the exemplary, non-binding case considered herein, $2^{100}/100\approx2^{93}$). Being able to satisfy a same piece request by disseminating, each time, different sets of DF blocks may enhance diversity of the information flowing over the network.

[0127] By adopting such an approach, piece requests PR can be met by the supplier SS by means of DF blocks flowing over the network that are all different and still represent the same piece.

[0128] Various embodiments may address the issue of how to "remap" the delivery of the DF blocks relating to a piece, that is, what parts of a whole DF sequence DFS may be used to satisfy two identical piece requests from one peer in the network.

[0129] Exemplary approaches adapted to be used in various embodiments may include:

[0130] a link to the network topology;

[0131] a pseudo-random choice.

[0132] A basic concept underlying the former approach (i.e., link to the network topology) is linking each DF section (i.e., a section of the DF sequence DFS) to a position of the node in the P2P network tree.

[0133] A relationship between DF block generation and network mapping may be the following: the number of hops H from the root of the tree structure (e.g., the initial seed SS, or possibly the tracker T) may represent information associated with the knowledge of the neighbors, e.g., as a result of the search that each peer performs when entering the P2P network. The maximum number R of receiver peers per seed or supplier may also provide related information.

[0134] A DF engine provided with these input parameters may thus identify a precise DF block in the DF sequence, i.e., may link the network topology with DF block generation based on certain rules.

[0135] An exemplary rule may involve identifying each DF section by at least one of the hop h from the tracker T and the index r for the receiver peer as schematically represented in FIG. **17**.

[0136] By way of non-limiting example, starting from a set of input parameters such as:

[0137] N_{DF}=# DF blocks per piece

[0138] R=Max receivers per seed

[0139] H=hops from root

[0140] r=[0, R−1] index receiver

[0141] k=[1, N_{DF}] index DF blocks

[0142] h=[0, 1, . . .] hops from tracker

[0143] the following return values:

[0144] i: index of the DF block in the whole DF sequence

[0145] s: index of the DF section

[0146] may be computed based on the following formulas

$$i=(h\cdot R+r)N_{DF}+k \quad \text{Eq. 1}$$

$$s=(h\cdot R+r) \quad \text{Eq. 2}$$

[0147] As schematically represented in FIG. **17**, Equation 1 can both shape the P2P tree and indicate to the DF generator of the seed the strategy to collect DF blocks from different DF

sections for each receiver peer in order to maximize the diversity of the information in the network.

[0148] An elementary tree network may stem from one seed. The BFDR procedure as exemplified herein will distribute the blocks with one root at the top of the network flow that starts from the seed and propagates through the rest of the nodes following Equation 1.

[0149] By way of simple explanation, one may refer to an exemplary scenario where R=2 and only one seed (for example the main server of a content broadcaster) is present. The root is at the level/hop '0'. The root has two "children", and each child has in turn two children:

[0150] Hop 0: 1 node;

[0151] Hop 1: 2 nodes;

[0152] Hop 2: 4 nodes . . . and so on . . .

[0153] Hop h: R^h nodes;

[0154] Equation 2 may identify the section just like Equation 1 may identify the block to be supplied by the peer identified by the couple (r, h) in the network map. In fact, it is sufficient to know the hop (the "distance from the top") and the receiver index (the "distance from the left") to locate the peer in the tree.

[0155] In various embodiments, the model and formulas above may permit identifying the relationship between the "parent" and the "children" sections.

[0156] For instance, one may assume that the integer index X identifies the DF section in the DF sequence that a generic peer is receiving. In that case, not the position of the peer in the tree but rather the position of the DF section received in the whole DF sequence may be of interest.

[0157] One may refer to the sections that, later on, will be supplied and received by the "children" as X_{chld}, where X_{chld} represents the integer index pointer in the DF sequence. In other words, while the section supplied may be identified using Equation 2, the section that the (child) peer will supply later on may be identified via the following formula:

$$(r \cdot X) \leq X_{chld} \leq (R \cdot X) + R - 1 \qquad \text{Eq. 3.}$$

[0158] With reference to the scheme just described, in various embodiments the possibility may exist of choosing the parameter R, that is the maximum number of receiver peers per supplier (seed) complying with certain conditions. That means, i.a., that each supplier/seed may have a value for R of its own, that is R_i.

[0159] Various embodiments may adopt variant solutions:

[0160] R may be decided arbitrarily by the developer via a predefined table;

[0161] R may be in relation with the number of hops via a generic function;

[0162] R may decrease while the tree gets larger in order to allow for the increase in size of the whole swarm;

[0163] R may increase with the number of hops, if bandwidth is available, to collect the contributions of the upload resources of each peer in the swarm and decrease the duration of the startup phase;

[0164] R may be a function of the upload bandwidth available at the supplier;

[0165] R may be a function of the download bandwidth available at the receiver peer; for instance, the receiver peer may have a download connection which is excessively limited because other services are running: so instead of 10 downloading sessions from suppliers at the same time, only 2-3 sessions may be accepted;

[0166] R may be dependent on event congestions revealed by external (i.e., "third") algorithms/engines;

[0167] R may be dependent on the type of service/feature desired by the receiver peer (i.e., end user); for instance in some embodiments, preference may be given to downloading in an ordered (chronological) fashion. If the bitstream downloaded represents video content, the end user may be interested in previewing the contents of the program just downloaded. Therefore, out of a DF piece related to the end of a movie and a DF piece at the beginning, the protocol may choose the latter in order to get a flowing video consumption without undesirable "freeze";

[0168] R may be dependent on the behavior of the peer; a table, ranking good and bad peers, may inhibit a supplier from initiating upload sessions to bad peers, thus giving an (at least first) preference to good peers;

[0169] a combination of the options above.

[0170] A mapping scheme generated by these exemplary rules may shape (or map) a P2P network as a logical tree, which is the dual-family scheme opposed to a mesh where contributions from the various peers follow a best-effort rule. While a mesh may be more robust than a tree in case of node failure or user disconnection, a tree may be more efficient, especially in the BFDR procedure exemplified herein, where information packets tend to be "pushed" rather than "pulled" i.e., requested.

[0171] The exemplary scheme considered as represented by Eqs. 1 and 2 may reveal that, as schematically portrayed in FIG. **17**, information may generated following Eq. 1 in a deterministic fashion at Hop H=1. In fact, the peer that has received the whole piece can create a "right" DF block from the proper DF section identified by the mapping scheme. If a peer had just received some block but not the whole piece, it is possible for it only to forward some blocks. At the beginning, the root (or seed) node has the whole piece, while the nodes at the 1^{st} hop can only retransmit, the nodes at the 2^{nd} hop benefit from the diversification that took place at the root.

[0172] Thus, in various embodiments, all the other nodes may receive the same information until the nodes of the 1^{st} hop have received the whole piece and become in turn seeds/suppliers. The scenario gradually improves once the nodes of the 2^{nd} hop have received the whole piece—and become seeds/suppliers—and so on for each hop of the P2P tree.

[0173] Stated otherwise, in various embodiments, Eq. 1 may be "respected" at the beginning only by the root node; as time elapses and pieces are fully downloaded, Eq. 1 will be increasingly "respected" also by more distant nodes from the root as these become in turn seeds/suppliers. This is a sort of radial or "spherical" propagation within an ever-improving scenario as nodes enter at different times in correspondence with different logical locations of the tree, so that after some time seeds are distributed over the network and not just concentrated at the root of the network. In various embodiments, this mechanism may benefit from diversification of information delivery realized at the DF generation module as well as from the different distribution of seeds and resources in the network, this being likely to be a real scenario for a network.

[0174] An exemplary mapping scheme as described previously may be suited to a broadcaster's needs, since a broadcaster may usually have an interest in being the actor who delivers the contents into the network, while the overlay may

be defined by one initial seed and a "forest" of e.g., Set Top Boxes that operate just as repeaters of the content streamed by the broadcaster server.

[0175] Especially when the content is not "live", certain Video On Demand scenarios for large CDN's may provide for the presence of more than one seed in the network: for instance, a user can connect after half an hour, or can be just interested in one part of a full content, for example in a sports section at the end of a news section; also, more than one seed server may be an option for large regions served by a same content provider. In general, an interest may exist in that plural seeds are involved in content distribution.

[0176] FIG. **18** provides a differentiated exemplary graphical representation of flows belonging to different sections of a whole DF sequence DFS, by highlighting the possibility for each peer to receive a same piece with different DF blocks.

[0177] For instance, in the non-binding exemplary delivery distribution scheme of FIG. **18**:

[0178] the number of blocks per DF section (or piece) is N_{DF}=100;

[0179] the maximum number of receivers R per seed is R=10. The delivery index i is indicated for each block sent to each peer.

[0180] Block X_i denotes the (only) block of the whole DF sequence and P_R represents the peer corresponding to:

[0181] a receiver of seed 1 that belongs to the 1^{st} seed group (sub-tree under seed 1);

[0182] a peer at Hop h=0 from a seed of the network; since R=10, then (if one assumes that indexes start from 0) P_R is P_9.

[0183] In various embodiments, the tracker T may trace (in a known manner) the number of seeds that share the content in P2P network, which makes it possible to manage the content distribution by forcing all blocks to be different to the receivers at one hop from each seed.

[0184] FIG. **18** also evidences that, through the supervision of the tracker T and Equation 1, all peers may share different information. Specifically, FIG. **18** exemplifies the behavior of embodiments in the case of two seeds and peers that share blocks at Hop=1 from the root of the tree.

[0185] This scenario may be extended to hops >1 from the root of the tree.

[0186] FIG. **19** evidences the possible effects of increasing the dimensions of a P2P network in an exemplary delivery distribution scheme where:

[0187] the number of blocks per DF section (or piece) is N_{DF}=100;

[0188] the maximum number of receivers R per seed is R=10.

[0189] By referring to Eqs. 1, 2, and 3, the peer $P0_0$ at hop h=0 from the seed of the network will receive the 1^{st} DF section that corresponds to the blocks in the range [1, 100], $P0_1$ will receive the 2^{nd} DF section that corresponds to the blocks in the range [101, 200], . . . $P0_9$ will receive the blocks in the range [901, 1000] and so on. When $P0_0$ becomes a seed, it dialogues with the tracker T and the XOR tree model is applied to the below sub-tree as indicated in Eq. 1. $P0_0$ at hop h=1 from the seed of the network, receives the $(R \cdot r)^{th}$ section that corresponds to the blocks in the range [1, 100].

[0190] In various embodiments, a P2P swarm may host not more than 100 nodes. If the corresponding tree is particularly complex (many hops) peers at the edge at the overlay may still receive the same information.

[0191] While the description of exemplary embodiments provided in the foregoing refers to a model that is ideally applicable to a tree scheme (wherein each node has multiple receivers but only one supplier per DF section), various embodiments may apply to situations where peers may have multiple suppliers that are forwarding the same DF section. In certain situations, e.g., in the case of medium-large networks, far peers may not yet have different blocks, especially at start-up of a large P2P sharing session.

[0192] FIG. **20** exemplifies a situation where, with the DF block index "i" identifying in a univocal manner a DF section, the peer P_i receives the same DF section from both P_u and P_v.

[0193] Sharing the blocks as soon as possible better exploits the available upload bandwidth; however, the fact that an external far peer (as schematically represented in FIG. **20**) may receive identical DF blocks related to the same DF section will limit the benefits deriving from a Digital Fountain (DF) arrangement, which aims at ensuring that—within the information spread or diffused in a large P2P sharing session—each block can actually contribute new information.

[0194] In various embodiments, this issue may be addressed by providing, in addition to "Have" messages (as discussed in connections with FIGS. **8** to **10**), "Receiving" messages which may be used to coordinate reception of, and requests for, identical DF sections.

[0195] A simple, real-life scenario is exemplified in FIG. **21**, where far nodes, e.g., P_i, might end up by collecting identical DF section blocks from a swarm SW at different download speeds. In the exemplary case of FIG. **21**, both P_v and P_u are connected to P_i, which may coordinate the request of blocks from different parts of the buffer (e.g., the swarm SW). Such an option may be somewhat paralleled to buffer-mapping exchange as provided for in certain P2P protocol clients for streaming or file-sharing applications. There, each node knows pieces or blocks that are available among his neighbors.

[0196] Various embodiments herein may not involve asking for a precise block index—as it happens in Bit Torrent (BT)—or sending blocks in a push fashion without any previous check: various embodiments make take advantage of the fact that it is still possible to parallelize requests to the peers P_v and P_u in order to improve download speed by resorting to a general rule of block request.

[0197] For instance, in an exemplary, non-binding scenario where the peer P_i asks P_v and P_u for different DF blocks belonging to the same DF section, such a rule may involve e.g., P_i asking:

[0198] from P_u—the DF blocks in the index group U, where U includes the set of integers $[u_1, u_2 \ldots u_i \ldots u_M]$, and

[0199] from P_v—the DF blocks in the index group V, where V includes the set of integers $[v_1, v_2 \ldots v_i \ldots v_M]$.

[0200] Various embodiments may involve any of the following:

[0201] i) U and V are sets of consecutive integers, i.e., u_1=1, u_2=2, . . . , u_i=i and v_1=M+1, v_2=M+2, . . . , v_i=M+i; in that case: v_M=u_M and v_i=M+u_i;

[0202] This case may be extended to more than two suppliers, where each set of indexes is (subsequent) next to the other. Also:

[0203] u_M, v_M may be selected arbitrarily by the developer based on heuristics concerning general network

infrastructure characteristics: the values selected may be constant or variable based on network communications conditions;

[0204] u_M, v_M may vary as a function of the number of available suppliers: for example, the more the suppliers, the smaller the average u_M, v_M;

[0205] u_M, v_M may vary in relation to the available bandwidth (known) per supplier P_v and P_u: for example, the higher the bandwidth B_M the greater u_M, v_M; also u_M, v_M may vary in case of network congestions or peer turnover;

[0206] ii) U includes all even integers while V includes the odd integers: i.e. $u_1=1$, $u_2=3$, . . . and $v_1=2$, $v_2=4$ . . . ;

[0207] iii) U and V are disconnected sets of random integers; such an option may be of interest as it may more backward compatible with earlier versions of P2P clients.

[0208] iv) P_i asks from P_v and P_u a DF (XOR) combination of the available blocks (i.e., a sort of re-encoding of DF blocks by other DF blocks);

[0209] any combination of the foregoing.

[0210] As indicated previously, while certain embodiments may adopt a link to the network topology, various embodiments may adopt a DF pseudo-random choice.

[0211] A rationale behind this latter option may lie in that, if the number of blocks N per piece is large enough (for example N>10 may already be a "good" number for diversity in swarms SW that count ~100 peers) the length of the whole DF sequence DFS makes it possible to assign a different DF section to each piece request without having to pay too much attention.

[0212] In embodiments of this kind, a link to the network topology (e.g., distance from the root) may no longer be necessary, while the request may play a role. Consequently, in various embodiments:

[0213] i) if requests of pieces are not monitored by a central tracker T:

[0214] a) the supplier SS may send the DF blocks from a section selected randomly from the whole DF sequence DFS;

[0215] b) the supplier SS may send DF blocks selected by picking them up randomly from the DF sequence DFS;

[0216] c) the supplier SS may send DF blocks each picked up from a different DF section selected randomly in the whole DF sequence DFS;

[0217] d) the relative index position of the DF block in the DF section may be incremented constantly but the DF section changes at each count;

[0218] e) an arbitrary index of section I_s may be defined following a linear algebra rule Y=aX, i.e., the 1st block chosen from DF section (1), the 2^{nd} block chosen from DF section (3), the 3^{rd} block chosen from DF section (5), and so on, by following a more complex generic rule Y=f(X);

[0219] f) the choice may be a function of, e.g., the ID of the peer that enters the network; the time spent by the same ID peer into the network; or the piece index in the file;

[0220] ii) if requests of pieces are monitored by a central tracker T:

[0221] the central tracker T may assign groups of DF sections for each potential supplier. Among each group of DF sections, suppliers can provide the DF blocks by following options like those just considered under (a), (b), (c), (d), (e) and (f) in the foregoing;

[0222] the same option (e) may be adopted, with the section index I_s chosen by the tracker through various methods or formulas.

[0223] Various embodiments may adopt any feasible combination of the options and sub-options considered in the foregoing.

[0224] In exploiting the DF mechanism exemplified herein in order to distribute blocks in a (e.g., P2P) swarm, various embodiments may provide for encoded blocks being shared early throughout the network.

[0225] Both delivery approaches discussed previously may prevent peers from selecting the same DF section and various embodiments may take care of successfully forwarding packets without generating duplicate DF blocks, so that the advantages of the embodiments may be exploited.

[0226] For instance, in scenarios where a reasonable confidence exists that every peer will encode its own pieces using a unique seed for the DF block generator, forwarding blocks as soon as a peer receives them may enhance data distribution and minimize the startup time of the system as a whole.

[0227] In certain contingencies, the mesh topology of a swarm SW may involve the presence of loops: FIG. **22** schematically represents an exemplary case where data duplication may occur. There, P_0 and P_1 are shown to collect encoded DF blocks from a seed supplier SS and forward them to any other peer they are in contact with (e.g., P_2). Being unaware that they share a common peer SS, they may both blindly forward their collected blocks to, say, P_2, thus sending the same information twice and wasting resources that could find better use. Continuing the forwarding scheme, P_2 could in turn mirror the blocks back to P_0 and P_1 (as they may appear both as potential peers in need of blocks to complete a piece), creating a dangerous loop that could easily lead the system to a halt. Even if a tree structure is adopted, such a peer topology may arise when its "leaves" decide to exchange received data.

[0228] In various embodiments, such a situation may be countered by using any of the exemplary forwarding schemes considered and by adopting various approaches to deal with duplicate blocks.

[0229] For instance, various embodiments may adopt a "forwarding halt" approach, where duplicate DF blocks are dealt with locally: as soon as a peer discovers that duplicate data is circulating in the swarm passing through its endpoints (i.e., that duplicate block data are being propagated there through), it will stop the forwarding process, effectively breaking the loop. This may be achieved by keeping track of the received DF section information associated with each packet, without undermining the back-compatibility to previous versions of BF/BT clients and without adding overhead to the DF block payload. Since that exemplary scheme may be applied once that data is received, there may still some duplicate DF blocks traversing the mesh, with their presence detected and controlled by each peer. In the example shown, P_2 may still receive duplicate data from P_0 and P_1, but it will not forward data back to them or to any other peer connected to it.

[0230] FIG. **23** is schematically representative of a "receiver list passing" approach, where the possibility of sending duplicate data is countered in the first place by creating a distributed rule set ensuring that each peer will not waste resources. In one possible approach, every encoded DF

block may include an additional header (Forwarding control field) H with a list of peers towards which a sender may wish to forward data: in that way, the receiver will be able to check if a connected peer has already taken advantage of the forwarding scheme and will not push useless data towards the swarm.

[0231] In the exemplary case of FIG. **23**, when P_1 forwards the blocks received from the seed to P_0 and P_2, it also adds their ID in the forwarding control field of the block. When P_0 receives the blocks, it already knows what peer(s) is/are to be excluded from the forwarding process. The additional header H gives the opportunity of avoiding the creation of duplicate data from the source, while making backward compatibility with previous versions of the protocol less immediate.

[0232] FIG. **24** is schematically representative of a "spanning tree" approach that is a sort of a mixed approach between the two approaches considered previously: instead of accepting duplicate data or avoiding it, various embodiments may regulate the flow of forwarded data on the mesh links in a spanning-tree fashion. A peer detecting duplicate data may choose a favorite supplier from which to receive data and notify the other peers with a signaling packet so that these may avoid supplying redundant blocks through specific links. In that way, an underlying tree topology is created between peers, and forwarding can proceed according to a push model. The criteria used to select a particular peer as the block supplier may, e.g., be similar to those described previously for establishing a link to the network topology. Using this forwarding scheme, some duplicate data may at first be present on the mesh, until the underlying tree topology is established: once this happens, the following blocks will be distributed without further waste of resources. To better cope with peer churn, various embodiments may introduce a weighting factor on each disabled link in order to prevent the exclusion of peers from the swarm.

[0233] For instance, in various embodiments, as exemplified in FIG. **24**, as soon as P_2 receives a duplicate block from P_0 or P_1, it can notify P_1, and ask P_1 to stop forwarding further data. The link between P_2 and P_1 will be virtually removed from the mesh and the following blocks supplied by the seed will reach P_2 by passing only through P_0.

[0234] FIG. **25** is generally representative of various embodiments wherein information content arranged in pieces including blocks of bits is distributed over a network including plural nodes/terminals SS, P_0, P_1, P_2, P_3, P_4, . . . at least one of which (i.e., SS) acts as a "seed supplier," i.e., a source of the pieces of information distributed. Plural seed nodes may exist as shown in FIG. **18**.

[0235] Due to the "cooperative" nature of the network (e.g., P2P), terminals may be configured to act as peer terminals with at least one first peer terminal (e.g., P_1 or P_2) sending the information to at least one second peer terminal (e.g., P_4). Various embodiments may provide receiving at the first peer terminal a set of blocks B of a corresponding piece of information (PE, from SS) and reconstructing the corresponding piece of information from the set of blocks received. The pieces of information distributed over the network are fountain encoded (see, e.g., **100** in FIGS. **11** and **12**) by XOR-ing the blocks in a piece, whereby a received piece is reconstructable (see, e.g., DFM in FIGS. **9** and **10**) from a combination of a corresponding set of linearly independent XOR-ed blocks (e.g., X_1, X_2, X_3, . . .). In various embodiments, the first peer terminal (e.g., P_1 or P_2) may start sending to the at least one second peer terminal (e.g., P_4) blocks from the set of blocks B it is receiving before the corresponding piece of information becomes reconstructable at the first peer terminal from said set of blocks, that is, before the whole set of blocks corresponding to the piece is received and the piece reconstructed and possibly subjected to an integrity check, such as, e.g., a hash integrity check at the first peer terminal. Fountain encoding the pieces of information distributed over the network may be by means of plural sets of linearly independent XOR-ed blocks (see, e.g., DF1, DF2, DF3, . . . in FIG. **12**) so that a same piece of information may be distributed to a plurality of peer terminals (e.g., from SS to P_1 or P_2 in FIG. **25**) by sending to the peer terminals in the plurality different sets of linearly independent XOR-ed blocks (see, e.g., X_{101}, X_{102}, X_{103} sent to P_1 and X_{201}, X_{202}, X_{203} sent to P_2), thus avoiding undue redundancy in the information further propagated to other peers, such as, e.g., P_4 in FIG. **25**, which receives different DF blocks X_{1101}, X_{1102}, X_{1103} from P_1 and X_{2001}, X_{2002}, X_{2003} from P_2, respectively.

[0236] The use of various embodiments may be detected by tracing packets using simple network freeware sniffers. While possibly relying on different technologies, various embodiments improve over the basic BitTorrent pattern, e.g., in terms of download time as measurable by a common stopwatch, by also detecting that the piece-integrity-check constraint has been broken. This may be by sniffing the traffic on a local machine running a BitFountain scheme with, e.g., popular free software (such as Ethereal, a free-ware for Windows or Unix-Linux) for a few minutes and detecting that blocks are shared to other IP addresses before a piece is completed. Also, in various embodiments, a number may indicate the index of the DF block in the whole DF sequence to specify the elements of the DF matrix that are set to one: i.e., what blocks have been XORed from the original source. Headers are exempt from encryption, which makes it possible to see how blocks are XORed. Additionally, changes over conventional solutions are accessible in the header, which is totally clear, not encrypted to the sniffer.

[0237] Various embodiments may be applied to Set Top Box technology while being otherwise adapted for other types of P2P content sharing.

[0238] Full working systems applied to file-sharing may use a Digital Fountain and a UDP and still be backward compatible with BitTorrent.

[0239] Performance evaluation of various embodiments shows superior performance in comparison to BitTorrent with increased piece size, leading to a possible improvement of more than 33% in download time. Such improved performance may be advantageous, e.g., in the area of Set Top Box technology (for P2P and content delivery engines) and, more generally, in encoding and data generation technology. Improved performance may be advantageous not just in P2P environments, but also in other scenarios of content delivery other than P2P.

[0240] Another field of use may be apps for smartphones: sports events, concerts, and any other events which may benefit from P2P technology in sharing MultiMedia (MM) contents related to an event attended by thousands of people.

[0241] Of course, without prejudice to the underlying principles of the disclosure, the details and the embodiments may vary, even appreciably, with respect to what has been described by way of example only, without departing from the scope of the disclosure.

[0242] From the foregoing it will be appreciated that, although specific embodiments have been described herein

for purposes of illustration, various modifications may be made without deviating from the spirit and scope of the disclosure. Furthermore, where an alternative is disclosed for a particular embodiment, this alternative may also apply to other embodiments even if not specifically stated.

1-11. (canceled)

12. An apparatus, comprising:
a receiver configured to receive a block of information;
a recoveror configured to recover a portion of content in response to the received block of information; and
a transmitter configured to send the received block of information before the recoveror recovers the portion of the content.

13. The apparatus of claim **12** wherein:
the content includes a file;
the portion of the content includes a piece of the file; and
the block of information corresponds to the piece of the file.

14. The apparatus of claim **12** wherein:
the content includes a file;
the portion of the content includes a piece of the file; and
the block of information includes one of multiple blocks of information that correspond to the piece of the file.

15. The apparatus of claim **12** wherein:
the content includes a file;
the portion of the content includes a piece of the file; and
the block of information includes a portion of the piece of the file.

16. The apparatus of claim **12** wherein:
the transmitter is configured to send a request for the portion of the content over a network; and
the receiver is configured to receive the block of information over the network.

17. The apparatus of claim **12** wherein the transmitter is configured to send the received block of information before the receiver has received a number of blocks of information, the number of blocks being sufficient for the recoverer to recover the portion of the content.

18. The apparatus of claim **12**, further comprising:
wherein the receiver is configured to receive other blocks of information that correspond to the portion of the content; and
the recoverer is configured to recover the portion of the content by exclusive-OR-ing groups of the received blocks to recover subportions of the portion of the content.

19. The apparatus of claim **12**, further comprising an encoder configured to encode the recovered portion of the content by exclusive-OR-ing blocks of the recovered portion to generate blocks of information that correspond to the recovered portion of the content.

20. The apparatus of claim **12** wherein the transmitter is configured to send an indication of a destination of the sent block of information.

21. A system, comprising:
a first peer configured to send a first block of information that corresponds to a portion of content;
a second peer configured
to receive the first block of information from the first peer,
to recover the portion of content in response to the first block of information; and
to forward the first block of information before recovering the portion of the content; and
a third peer configured
to receive the first block of information from the second peer; and
to recover the portion of the content in response to the first block of information.

22. The system of claim **21** wherein:
the second peer is configured to send a request for the portion of the content to the first peer;
the first peer is configured to send the first block of information in response to the request from the second peer;
the third peer is configured to send a request for the portion of the content to the second peer; and
the second peer is configured to send the first block of information in response to the request from the third peer.

23. The system of claim **21** wherein the second peer is configured to send the first block of information before having information sufficient to recover the portion of the content.

24. The system of claim **21** wherein the first peer is configured to generate the first block of information by encoding a subportion of the portion of the content.

25. The system of claim **21** wherein the second peer is configured:
to generate a header identifying the third peer as a destination of the first block of information; and
to send the header with the first block of information.

26. The system of claim **21** wherein the third peer is configured to request that the first peer send to the third peer a second block of information that corresponds to the portion of the content.

27. The system of claim **21**, further comprising:
a fourth peer; and
wherein the third peer is configured to request that the fourth peer send to the third peer a second block of information that corresponds to the portion of the content.

28. A method, comprising:
receiving with a first network node a block of information; and
sending the block of information from the first network node to a second network node before the first network node recovers a portion of content in response to the block of information, and before the first network node verifies the recovered portion of the content.

29. The method of claim **28**, further comprising:
recovering with the first network node the portion of the content in response to the block of information; and
after recovering the portion of the content, sending to at least one network node a request that no more blocks of information corresponding to the portion of the content be sent to the first network node.

30. The method of claim **28**, further comprising:
recovering with the first network node the portion of the content in response to the block of information; and
wherein sending the block of information from the first network node to the second network node includes sending the block of information after recovering the portion of the content but before verifying the recovered portion of the content.

31. The method of claim **28**, further comprising:
recovering with the first network node the portion of the content in response to the block of information; and

wherein sending the block of information from the first network node to the second network node includes sending the block of information before recovering the portion of the content.

32. The method of claim **28**, further comprising:

recovering with the first network node the portion of the content in response to the block of information;

encoding with the first network node the portion of the content into a first set of blocks of information and into a second set of blocks of information; and

sending with the first network node to a third network node at least one block from the first set of blocks; and

sending with the first network node to a fourth network node at least one block from the second set of blocks.

33. A non-transitory computer-readable medium storing instructions that, when executed by one or more computing apparatuses, cause at least one of the computing apparatuses, or at least one other apparatus under control of at least one of the computing apparatuses:

to cause a first network node to receive a block of information; and

to cause the first network node to send the block of information to a second network node before the first network node recovers a portion of content in response to the block of information, and before the first network node verifies the recovered portion of the content.

* * * * *