Coding Approach for Fault-Tolerance in Multiagent Systems

Filip Miletić, Patrick Dewilde
Faculty of Electrical Engineering, Mathematics and Informatics
Delft University of Technology, Mekelweg 4, 2628CD Delft, The Netherlands
tel: +31 15 278 1372, {f.miletic,p.dewilde}@ewi.tudelft.nl

Abstract—*When deploying a multiagent system in chaotic environments, some form of built-in fault-tolerance is a must. Fault-tolerance includes keeping backup copies of critical data. Existing platforms often choose to distribute data copies by replication. For most purposes, replication uses storage and bandwidth inefficiently. We investigate an information theoretic approach to choosing the most effective way to distribute this data. This approach uses erasure coding as an efficient way to avoid unneeded replication and still yield good error recovery. We propose the* erasure-graph *model of the multiagent society. This model describes the interconnections in the society, considering also ways the links and agents can fail. We give a partitioning schema agents can use to find the best distribution, given the knowledge of the erasure-graph for the society. Computing the partition is easily distributed over the agent society.*

1 INTRODUCTION

Agent programs in multiagent systems (MAS) often use an *agent platform* as middleware [11]. The platform controls the messaging and the agent life-cycle. The reliability of the agent platform has received little attention *in practice*, though it is recognized as a MAS issue [22]. Notable MAS, such as JADE [3] or COUGAAR [1], assume a reliable platform. Sometimes the platforms provide persistence to non-volatile media, protecting the agents from transient failures. For both platforms, replication services exist; these services have been implemented with simplicity in mind, rather than efficiency. In this paper we user information-theoretic arguments to show how efficient reliability in a MAS can be derived from cooperation with a slight trade-off for an increase in complexity. The resulting platform is called Combined; this platform is based on the COUGAAR agent architecture, with additions that make it suitable for deployment in chaotic, rapidly changing environments.

1.1 Application

The main motivation for looking at increasing the reliability of multi-agent platforms is because it is desirable in applications. We will mention two applications which benefit from the Combined approach. Emergency search-and-rescue operations benefit from timely information delivery. This deployment setting was also adopted as the prototype application for Combined. We envision a near-future application in which members of emergency service units (police, fire brigade, ambulance etc.) are equipped with short range communication devices. Due to the nature of the deployment, it cannot assume that communication infrastructure exists; the reliability of the communication devices can also not be taken for granted. The devices work around these impediments by cooperating in message delivery and keeping each other's observations; the body of the paper explains the theory on which this possibility is based. As another example, large scale distributed scientific applications, such as Seti@Home [21], or Folding@Home [6] succeeded in harnessing worldwide processing power for highly demanding computational tasks. The focus of their system architectures has shifted from traditional parallel and distributed computing efficiency issues [12] to organizing and managing redundant work done by worldwide computers. Some indication of the system architecture can be found at [2]. The Combined approach might aid this effort by offering a way to preserve computation results despite computers leaving the network.

1.2 Related Work

The system outlined in the introduction naturally compares itself to similar peer-to-peer (P2P) systems. The early P2P systems such as Napster [17], Gnutella [8], FreeNet [7] and MojoNation [15] were the first to introduce the promise

KIMAS 2005, April 18-21, 2005, Waltham, MA, USA.
0-7803-9013-X/05/$20.00 © 2005 IEEE.

of ever-present file storage, and were a motivation for this work. Subsequent P2P systems, such as Tapestry [24], Chord [16], Pastry [20], Kademlia [13], Content Addressable Network [18] discuss efficient key-based routing in various settings. These methods support deterministic message routing in an overlay network. OceanStore [10] is an architecture for establishing global persistent storage; similarly, the Cooperative Filesystem (CFS) is an application that uses Chord to make a P2P-based read only storage system. Unlike Combined, all these systems work with an overlay on top of an existing Internet Protocol (IP). The underlying IP network layer allows contact between any pair of the participants at low cost. The interconnect mesh corresponds to a complete graph. In Combined, however, connection establishes a spatial proximity based network mesh, with links induced by the distance between nodes. Due to this external constraint, the interconnect mesh is much sparser than that of the mentioned systems. The contact between far away nodes can be costly. Typically, the transitive closure of the Combined interconnect mesh would correspond to the IP-based interconnect used by the mentioned P2P systems. Another important distinctions is that in Combined, cooperation is needed for even the simplest operations such as contacting a far away node. In the mentioned P2P systems, this operation is handled by a lower level network protocol. The layering simplifies the design, but limits the applicability of the overlay to networks in which IP is efficient. The ideas of Combined are most thoroughly shared with OceanStore. This architecture uses erasure coding to achieve high data reliability. The improvement from using erasure coding over replication was clearly shown in another paper [23] from the co-authors of OceanStore. OceanStore considers erasure coding only for deep archival; but also goes to considerable length to handle data security. Combined intends to use data archival to provide process persistence; but it does not consider data security explicitly.

1.3 Result

In Section 2, we present a graph-theoretic model using an *erasure graph* to describe the system interconnections, and a *partition encoding* for allocating data pieces (tags) to different agents. We then use this model to derive the storage capacity of the resulting society, and prove that the capacity cannot be larger than the availability figure of the most reliable agent in the network (Section 2.1). Assuming the availabilities are known, we describe the strategy that an agent can use to determine which code and data partition to choose to make best use of the available capacity (Section 2.2). We first give a calculation for a general interconnection case, and random agent position within the network. We then simplify the analysis by layering the neighbours in tiers and allowing agents to delegate pieces of work to each other.

2 SYSTEM MODEL

We now give an overview of the system setup and notation. For detailed definitions and more flexible models, the reader is referred to [14]. The network of nodes is constructed based on the interconnections created by a short range radio network: two nodes are connected if they are closer to each other than a pre-determined *range*. This is a reasonable assumption, shared by papers closely dealing with wireless network properties, such as [9]. Denote as V the set of nodes $\{v_1, \ldots, v_m\}$. If a connection between two different nodes v_i and v_j exists, we insert the pair $\{v_i, v_j\}$ into the set E. The graph $G(V, E)$ is interpreted in the usual sense: V is a set of vertices, and E is a set of corresponding edges. When we need to emphasize that V or E belong to a graph G, we write V_G and E_G, respectively. The interconnection mesh of G changes as the nodes move, join or leave the network. When nodes leave the network, it can happen either *voluntarily* or *involuntarily*; the latter can occur because of network partitions or nodes being damaged, out of power, or destroyed. The involuntary leave is called an *erasure*. Due to nodes leaving the network, the interconnection graph *degrades*. If a node v leaves, it is removed from V, along with the corresponding edges from E. Here we assume that each node can leave independently with identical probability ε. We call the graph G the *erasure graph*. Consider a *source node* v from V that has during its lifetime produced valuable data (*a tag*) that should be preserved in case that v gets erased. Node v can choose to deposit copies of the tag with its neighbours. Depositing copies is commonly called *replication*. This approach, although effective, uses too much excess storage. Moving tags also costs bandwidth, so a smart source node would want to minimize the cost. In [23], an alternative approach was discussed, whereby the node uses an erasure code to transform the tag and deposits only fragments of the resulting coded messages to neighbours. Thanks to the coding, only a fraction of the fragments is required for successful decoding. There exist efficient methods (for instance, the iterative decoding method from [19]) to recover original tags. OceanStore and CFS [5] use these methods, but assume that the missing tags were erased independently, and those that remain are accessible independently. In the locally-connected network mesh, these assumptions do not necessarily hold. When trying to deposit a tag, v will have a choice of contracting several neighbours and request them to be tag *keepers*. Typically, v will want to pre-code the tag as said before copying to the keepers. v would then make a *partition* of the entire message. The partition is described by a set $\Pi = \{\pi_1, \ldots, \pi_m\}$. Each element of the partition is in itself a set denoting which bit of the encoded message is allocated to which node. For example, if the message was 5 bits long, and 2 nodes were available, one possible partition would be: $\Pi = \{\pi_1, \pi_2\}$, where $\pi_1 = \{1, 2\}$ and $\pi_2 = \{3, 4, 5\}$. This means that bits 1 and 2 are allocated to the node with index 1, and bits 3, 4 and 5 are allocated to the node with index 2.

To store the fragments, the source node v chooses the code and the partition. It then contacts the nodes, as defined by the partition, and deposits the message fragments. This is called a *write*. To retrieve the stored information, v contacts all the available keepers and retrieves the fragments. The fragments are then re-assembled and decoded to produce the original tag. This is called a *read*. Between the writes and corresponding reads, the keepers may be erased, or may be unavailable at the time of the read. Although each keeper has a probability ε to disappear, its availability also depends on its connection to other nodes. We will denote the availability of the node v_i from V as w_i. The writer will typically want to know in advance the availability of the potential keepers to decide which partition is the "best". Due to erasures, the graph G could be partitioned into connected components. We denote as $\mathcal{Q}_G$ the set of all maximally connected components for a given graph G. These maximal connected components are called *chains*. Degrading a graph is called a *transition*. Two simple writer decisions are given in the following examples. The decisions are named according to the cooperation policy: the *Lone Ranger* prefers to keep all the message to itself; the *Cloner* prefers to make multiple identical copies of the message.

Example 1 (The Lone Ranger) *The source decides to keep the entire message on a single node. The probability that the message is readable is:* $\Pr(\textit{readable}) = \Pr(\textit{keeper alive}) \cdot \Pr(\textit{path to keeper exists}) \leq \Pr(\textit{keeper alive}) = \varepsilon$.

Example 2 (The Cloner) *The source node decides to duplicate the tag with k keepers. This ensures that if at least one keeper is present, the entire tag can be retrieved. However, the case in which such storage space investment is justified is very improbable. The probability that all but one keeper are absent is: $k(1-\varepsilon)\varepsilon^{k-1} \leq k\varepsilon^{k-1}$, and tends exponentially fast to zero with k.*

2.1 The Capacity of the Erasure Graph

For a given connected graph G, a writer would like to know the storage capacity that G can offer. This section provides tools to do just that. Let us denote as X the message that a writer needs to distribute across the network formed by nodes V. For convenience, we consider that X is a string of n binary digits. The reader will be able to retrieve only some fragments, due to the degradation of the network. The mutual information[1] on X and Y is given as: $I(X,Y) = H(X) - H(X|Y)$, where $H(X)$ and $H(X|Y)$ are, in order, the entropy of a random variable X and the conditional entropy of a random variable X given Y [4]. Since for a known X, any Y is known, it holds that: $I(X,Y) = H(Y) - H(Y|X) = H(Y)$, i.e., knowing X gives all the information over Y. The writer has some freedom to choose the partition Π, if it knows the availabilities for the nodes v_i from V. This way, $H(X|Y)$ becomes a function of Π. Knowing all w_i, writer can choose the Π that maximizes $H(X|Y)$. The capacity of G is then $C = \max_\Pi I(X,Y)$. We now express C in terms of w_i in Theorem 1, and find C explicitly in Theorem 2.

Theorem 1 (The Capacity of G given Π) *Let G be an erasure graph. The capacity of the channel defined on the graph G, under a given partition encoding Π and assuming uniform connection probability, is obtained by solving a linear program:*

$$C = \max_{\Pi} \sum_{1\leq i\leq |V|} w_i|\pi_i|; \sum_{1\leq i\leq |V|} |\pi_i| = n \qquad (1)$$

where coefficients w_i for $1 \leq i \leq |V|$ depend on the connectivity of the graph G:

$$w_i = \sum_{e=0}^{|V|-1} \frac{\varepsilon^e(1-\varepsilon)^{|V|-e}}{|V|-e} \cdot \cdot \sum_{E_p\in\mathcal{P}_e} \sum_{H\in\mathcal{Q}_{E_p\circ G}} |V_H|\iota(v_i \in V_H). \qquad (2)$$

In Equation (2), $\iota(x)$ is an indicator function, equal to 1 when x is true, and equal to 0 otherwise. E_p is the *erasure pattern*, a function mapping each vertex of G into the set $\{0,1\}$ and associated with a particular transition. It maps $v \in V$ to 1 if v is erased by the transition, or to 0 if v is not erased by the transition. E_p can be *applied* to $\mathcal{Q}_G$, to obtain a new connected components set: each vertex that E_p maps to 1 is removed from G along with corresponding edges. $\mathcal{Q}_{E_p\circ G}$ is the maximally connected component set that G is split into after applying the erasure pattern E_p. $\mathcal{P}_e$ is the set of all erasure patterns on G, having exactly e erasures. A consequence of Theorem 1 is the maximum capacity obtainable for a given graph G.

Theorem 2 (The Maximum Capacity of G) *Let the channel be defined on the graph G, and let all availabilities w_i be known for all nodes v_i from V. Let $w^* = \max_{1\leq i\leq |V|} w_i$. Then it holds that: $C = nw^*$.*

2.2 The Choice of the Partition

We have seen how the node availabilities affect the storage capacity achievable by a given graph G. We will now describe the strategy the writer uses to find the "best" partition. In [14], the subset of acceptable partitions is captured by the following definition.

[1]The meaning of mutual information in this case is simply the number of bits shared by X and Y. Likewise, the entropy $H(X)$ is the number of bits in X, and the conditional entropy $H(X|Y)$ is the number of bits that are left unknown in X if we know all the bits of Y.

Definition 1 (Distortion) *Consider the transition between the graphs with corresponding maximally connected component sets $\mathcal{Q}_G$ and $\mathcal{Q}_{E_p \circ G}$. Let D_c be the set of all possible connected component configurations obtainable by applying an erasure pattern E_p to $\mathcal{Q}_G$, and let $V(\mathcal{Q}_{E_p \circ G})$ be the set of all vertices thereof. Let $R = (1-\varepsilon_T)n - \sum_{i \in V(\mathcal{Q}_{E_p \circ G})} |\pi_i|$. The* distortion *is given by:*

$$d(\mathcal{Q}_G) = \sum_{\mathcal{Q}_{E_p \circ G} \in D_c} \Pr(E_p) \cdot R \cdot \iota(R > 0). \quad (3)$$

Given a maximum distortion $d_{\max}$ and the area $v(d_{\max}) = \{\mathcal{Q}_{E_p \circ G} | d(\mathcal{Q}_{E_p \circ G}) \leq d_{\max}\}$, the optimal configuration $\mathcal{Q}_G^*$ is given by

$$\mathcal{Q}_G^* = \arg \min_{\mathcal{Q} \in v(d_{\max})} f(\mathcal{Q}), \quad (4)$$

where f is a disambiguation function that helps in the choice of the unique solution. The parameter ε_T is the *decoding threshold* [19] of the employed decoder, the fraction of n that can be erased, without entailing decoding error. f is a goal function picked according to design criteria: (i) Given a threshold ε_T, decoding must succeed if erasure fraction is less; (ii) Given two partitions from $v(d_{\max})$, the encoder chooses a "more distributed" one; (iii) The encoder must handle a variable number of hosts ($|V|$), and variable message lengths n. The requirement (i) is implicitly satisfied by choosing a threshold-ε_T code. Further requirements can be satisfied in different ways; we choose the partition Π such that the sum-squared of all fragment lengths ($n_i = |\pi_i|$) is minimal. This choice is akin to mean-square energy minimization in multi-channel signal detection : we assume that each fragment length contributes independently to the entire tag. The expected number of retrieved bits must be equal to $(1-\varepsilon_T)n$. This expression is precisely equal to the sum of n_i weighed by w_i for each node. Finally, the fragments should form a partition of the message, thus the sum of n_i must be equal to n. For the same reason, for all i, the condition $n_i \geq 0$ must hold. This argument gives rise to Theorem 3.

Theorem 3 (Choice of the Partition) *Let n be the length of the message, let w_i be the availabilities of all the nodes from V, for $1 \leq j < i \leq |V|$ and assume $0 < w_j < w_i < 1$. Assuming the least-squares disambiguation f, the partition n_i, is given:*

$$\begin{aligned} n_i &= \lceil \lambda_1 w_i + \lambda_2 \rceil, \\ \lambda_1 &= (nS_w + |V|n(\varepsilon_T - 1))/(S_w^2 - |V|S_{w^2}), \\ \lambda_2 &= -(nS_{w^2} + n(\varepsilon_T - 1)S_w)/(S_w^2 - |V|S_{w^2}), \end{aligned} \quad (5)$$

under the code feasibility *condition* $\min_i w_i < 1 - \varepsilon_T < \max_i w_i$.

As some n_i may come out to be negative, violating condition $n_i \geq 0$, we settle for the point closest to the one found, but which lies in the area $n_i \leq 0$. In equation 5, S_w and S_{w^2} are shorthands for $\sum_i w_i$, and $\sum_i w_i^2$ respectively. The writer can use the code feasibility condition to choose the code: if $\max_i w_i < 1 - \varepsilon_T$, the code is *unfeasible*; if $1 - \varepsilon_T < \min_i w_i$, the code is *(over)feasible*. The Cloner strategy is always *(over)feasible*, so there always exists at least one applicable strategy for the writer. To compute the partition, the writer needs the tuple: $T_V = \left(\sum_i 1, \sum_i w_i, \sum_i w_i^2, \min_i w_i, \max_i w_i\right)$. By partitioning V into $V_1, \ldots V_l$, the writer can compute n_i by combining the results obtained from each T_{V_l}. The writer will sub-divide V into subsets depending on which its immediate neighbour they can be reached by (see Figure 1). It will then deliver to each V_l the fragments for all the members of T_{V_l}, along with its computed value of λ_1 and λ_2. Each V_l will have enough information for further subdivision and delivery.

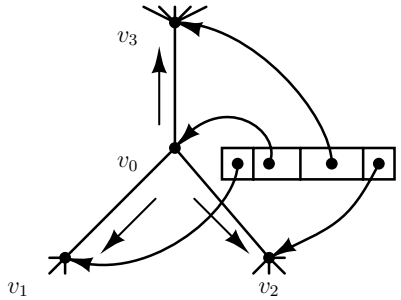

Figure 1: Subdividing V into subsets headed by nearest neighbours. v_0 is the writer. It delegates 3 fragments of a single tag to its nearest neighbours, and v_3. v_0 also delivers λ_1 and λ_2 to each of the neighbours.

3 CONCLUSIONS

This paper presented an information-theoretic approach to improving multiagent platform reliability. We show how, by using coding and partitioning, it is possible to achieve reliable data storage even when the platform itself is unreliable. We define a criterion for a feasible code that is used to choose the coding and partitioning. We show how the writers can then make first partitions, and then delegate their neighbours with sub-partitions. These information-theoretic arguments come from the vast information and coding theory literature but have so far received comparatively little attention in multiagent platforms despite potential gains. Multi-agent platforms that employ this (or similar) coding types can be made reliable enough to use in adverse, chaotic environments. The exposition in this text assumes that the availability estimates are known for all the nodes. In practice the estimates may be costly to obtain. It may be acceptable that a lower bound on the availability estimates is substituted. These can be obtained at comparatively little cost, by sub-dividing the nodes based upon the proximity to the writer, and then considering only a subset of possible ways that a node can be accessible to the writer. Apart from yielding the availability estimates,

the sub-division can give rise to a distributed control algorithm, whereby once determined, the partition is allowed to dynamically change to compensate for changes in the network connectivity. This algorithm and its properties are recommended for further work. The research has been funded by DECIS Lab, Delft as a part of the Project Combined.

REFERENCES

[1] BBN Technologies. *The Cougaar Architecture Guide*, 2004.

[2] Boinc: Berkeley Open Infrastructure for Network Computing. Online reference. http://boinc.berkeley.edu/.

[3] Elisabetta Cortese, Filippo Quarta, and Giosue Vitaglione. Scalability and performance of JADE message transport system.

[4] Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory*. Wiley Series in Telecommunications. John Wiley & Sons, New York, NY, USA, 1991.

[5] Frank Dabek, M. Frans Kaashoek, David Karger, Robert Morris, and Ion Stoica. Wide-area cooperative storage with CFS. In *Proceedings of the 18th ACM Symposium on Operating Systems Principles (SOSP '01)*, Chateau Lake Louise, Banff, Canada, October 2001.

[6] Foldinghome. Online reference. http://folding.stanford.edu.

[7] Freenet. Online reference. http://freenet.sf.net/.

[8] Gnutella. Online reference. http://gnutella.wego.com/.

[9] P. Gupta and P. Kumar. Capacity of wireless networks, 1999.

[10] John Kubiatowicz, David Bindel, Yan Chen, Patrick Eaton, Dennis Geels, Ramakrishna Gummadi, Sean Rhea, Hakim Weatherspoon, Westly Weimer, Christopher Wells, and Ben Zhao. Oceanstore: An architecture for global-scale persistent storage. In *Proceedings of ACM ASPLOS*. ACM, November 2000.

[11] Luck et al. *Agent Technology: Enabling Next Generation Computing*. The Agentlink Community, 2002.

[12] Nancy A. Lynch. *Distributed Algorithms*. Morgan Kaufmann, 1996. LYN n 96:1 P-Ex.

[13] P. Maymounkov and D. Mazieres. Kademlia: A peer-to-peer information system based on the xor metric, 2002.

[14] Filip Miletić and Patrick Dewilde. Distributed coding in multiagent systems. In *IEEE Conference on Systems, Man and Cybernetics*. IEEE, October 2004.

[15] Mojonation. Online reference. http://www.mojonation.net/.

[16] Robert Morris, David Karger, Frans Kaashoek, and Hari Balakrishnan. "chord: A scalable peer-to-peer lookup service for internet applications". In *ACM SIGCOMM 2001*, San Diego, USA, September 2001.

[17] Napster. Online reference. http://www.napster.com/.

[18] Sylvia Ratnasamy, Paul Francis, Mark Handley, Richard Karp, and Scott Shenker. A scalable content addressable network. Technical Report TR-00-010, University of California at Berkeley, Berkeley, CA, 2000.

[19] T. Richardson and R. Urbanke. Modern coding theory, June 2003.

[20] Antony Rowstron and Peter Druschel. Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems. *Lecture Notes in Computer Science*, 2218:329–??, 2001.

[21] Setihome. Online reference. http://setiathome.ssl.berkeley.edu/.

[22] V. S. Subrahmanian, Piero Bonatti, Jürgen Dix, Thomas Eiter, Sarit Kraus, Fatma Ozcan, and Robert Ross. *Heterogeneous Agent Systems*, chapter 1, page 21. MIT Press/AAAI Press, Cambridge, MA, USA, 2000.

[23] H. Weatherspoon and J. Kubiatowicz. Erasure coding vs. replication: A quantitative comparison.

[24] Ben Y. Zhao, Ling Huang, Jeremy Stribling, Sean C. Rhea, Anthony D. Joseph, and John D. Kubiatowicz. Tapestry: A resilient global-scale overlay for service deployment. *IEEE Journal on Selected Areas in Communications*, 22(1):41–53, January 2004.

A DERIVATIONS AND PROOFS

Proof. (Theorem 1) The probability of e erasures on a single transition is given by:

$$\Pr\left(e \text{ erasures}\right) = \binom{|V|}{e} \varepsilon^e (1-\varepsilon)^{|V|-e}.$$

Generate the set of all possible erasure patterns for a given e and name it $\mathcal{P}_e$. Every $E_p \in \mathcal{P}_e$ induces a set of connected components $\mathcal{Q}_{E_p \circ G}$, and each of the sets is composed by connected graphs $G_{Ei} \in \mathcal{Q}_{E_p \circ G}$. The mutual information $I\left(X,Y\right)$ is the mean of the number of bits retrieved from all the possible erasure patterns. The set of all possible erasure patterns $\mathcal{P}$ is given by the union of all individual erasure patterns: $\mathcal{P} = \bigcup_{0 \leq e \leq |V|} \mathcal{P}_e$. The average number of bits retrievable from the channel depends on the particular

transition, and ultimately of the erasure pattern within. Considering e known and looking at a particular erasure pattern E_p, the average number of bits is obtained by averaging over all possible connection points, since by assumption of the theorem, a reader can be with uniform probability connected to either of the remaining nodes of $\mathcal{G}$.

$$I\left(X,Y|\mathcal{Q}_{E_p\circ G}\right)=\frac{\sum_{G\in\mathcal{Q}_{E_p\circ G}}|V_G|I\left(X,V_G\right)}{\sum_{G\in\mathcal{Q}_{E_p\circ G}}|V_G|}. \quad (6)$$

From here it is easy to obtain that: $I\left(X,Y\right) = \mathrm{E}\left[I\left(X,Y|\mathcal{Q}_{E_p\circ G}\right)\right]$ will yield the expression for mutual information for X and Y. These expressions depend upon elements of Π. The expression includes multi-dimensional sums, which ultimately depend on the (beforehand unknown) connectivity of graph G. Let e remain fixed, and let us focus on equation (6). For a given $v \in V$, define $\langle v\rangle|$ to be the index of v. Substituting $I\left(X,V_G\right)$ by $\sum_v \in V_G|\pi_{\langle v\rangle}|$, it is seen that $|\pi_{\langle v\rangle}|$ enters the sum on a number of occasions. It is possible to determine how many times and with what weight coefficient does $|\pi_{\langle v\rangle}|$ appear in the appropriate equation, and the answer depends on e and on the size and number of the connected components that v can belong to. Let the connected components be named *chains*, for convenience. The shortest chain that v can be part of has size 1, when v is its only element. The longest chain has size $|V|$, when no degradation takes place. If v belongs to a chain of size l, when e errors are present, its contribution to $I\left(X,Y\right)$ depends on the position of v in the string. There are different ways in which v can be a member of a chain of size l. Expanding the expectation in $I\left(X,Y\right)$, one obtains

$$\begin{aligned} I\left(X,Y\right) &= \mathrm{E}\left[I\left(X,Y|\mathcal{Q}_{E_p\circ G}\right)\right] \\ &= \sum_{E_p\in\mathcal{P}}\frac{\sum_{G\in\mathcal{Q}_{E_p\circ G}}|V_G|I\left(X,V_G\right)}{\sum_{G\in\mathcal{Q}_{E_p\circ G}}|V_G|}\cdot\Pr\left(\mathcal{Q}_{E_p\circ G}\right) \quad (7)\end{aligned}$$

and averaging the number of bits retrieved from each element of $\mathcal{P}$ will give $I\left(X,Y\right)$. Since the degradation model is i.i.d., the probability of each $E_p \in \mathcal{P}_e$ is equal to $\varepsilon^e(1-\varepsilon)^{|V|-e}$. By introducing an indicator function it is possible to restate $I\left(X,V_G\right)$ as:

$$\begin{aligned} I\left(X,V_G\right) &= \sum_{v\in V_G} I\left(X,v\right) \\ &= \sum_{i=1}^{|V|} I\left(X,v_i\right)\iota\left(v_i\in V_G\right) \\ &= \sum_{i=1}^{|V|}|\pi_i|\iota\left(v_i\in V_G\right). \quad (8)\end{aligned}$$

which permits separation of the sum in equation (6) as:

$$\begin{aligned} I\left(X,Y\right) &= \sum_{E_p\in\mathcal{P}}\frac{\Pr\left(\mathcal{Q}_{E_p\circ G}\right)}{\sum_{G\in\mathcal{Q}_{E_p\circ G}}|V_G|}\cdot \\ &\quad\cdot\sum_{G\in\mathcal{Q}_{E_p\circ G}}|V_G|\sum_{i=1}^{|V|}|\pi_i|\iota\left(v_i\in V_G\right). \quad (9)\end{aligned}$$

Substituting $\mathcal{P}$ this becomes:

$$\begin{aligned} I\left(X,Y\right) &= \sum_{E_p\in\bigcup_{0\le e\le|V|}\mathcal{P}_e}\frac{\varepsilon^e(1-\varepsilon)^{|V|-e}}{\sum_{G\in\mathcal{Q}_{E_p\circ G}}|V_G|}\cdot \\ &\quad\cdot\sum_{G\in\mathcal{Q}_{E_p\circ G}}|V_G|\sum_{i=1}^{|V|}|\pi_i|\iota\left(v_i\in V_G\right). \quad (10)\end{aligned}$$

for which, when E_p is an element of $\mathcal{P}_e$, the following expression holds: $\sum_{G\in\mathcal{Q}_{E_p\circ G}}|V_G| = |V| - e$, which is easily seen to be true, since $\sum_{G\in\mathcal{Q}_{E_p\circ G}}|V_G|$ is the total number of nodes present in the connected component $\mathcal{Q}_{E_p\circ G}$. By changing the order of summation in equation (7) such that the first sum goes over all π_i, and noting that for $e = |V|$ the sum component is equal to zero, one is able to find: $I\left(X,Y\right) = \sum_{i=1}^{|V|} w_i|\pi_i|$, where w_i is given by equation (2). □

Proof. (Theorem 2) Let w_i and π_i be permuted, without loss of generality, so that $w^* = w_1 \ge w_2 \ge \cdots \ge w_{|V|}$.

$$C=\max_{\Pi}\sum_{1\le i\le|V|}w_i|\pi_i|\le w_1\sum_{1\le i\le|V|}|\pi_i|=w_1 n, \quad (11)$$

since for each i, $w_i|\pi_i| \le w_1|\pi_1|$. □

Proof. (Theorem 3) Let V be the set of nodes and $1 \le i \le |V|$, wherever the index i appears. Assume that all w_i are known and adopt a shorthand $n_i = |\pi_i|$. The minimization problem is: Minimize $\sum_i n_i^2$, given $\sum_i w_i n_i = (1-\varepsilon_T)n$, $\sum_i n_i = n$, while $n_i \ge 0$. First we solve the said constrained minimization problem. Then, we discuss the conditions under which the solution given by the partial problem fits all the conditions $n_i \ge 0$. By using the Lagrange multiplier method, we obtain the goal function: $F(n_i) = \sum_i \left(n_i^2 - \lambda_1 w_i n_i - \lambda_2 n_i\right)$. Taking partial derivatives of $F(n_i)$ for all n_i and solving the resulting system of equations for λ_1 and λ_2 (an exercise that we omit here), one obtains the equation 5. As n_i might not always be integer, we take the ceiling of the obtained value. To show the condition for the existence of positive solutions, we consider hyper-planes $H_1 : \sum_i w_i n_i = (1-\varepsilon_T)n$ and $H_2 : \sum_i n_i = n$. H_1 intersects the coordinate axes at points having coordinates all coordinates equal to 0, except for coordinate i, which is equal to $(1-\varepsilon_T)n/w_i$. Likewise, H_2 intersects coordinate axes at points whose i-th coordinate is equal to n. $\min_i w_i$ and $\max_i w_i$ always exist. Let w_- be the minimum, and w_+ be the maximum. The non-zero coordinates of H_1's intersection with the coordinate axes are then $(1-\varepsilon_T)n/w_- > n$ and $(1-\varepsilon_T)n/w_+ < n$, respectively. Consider the plane α determined by these two points and the coordinate system origin. The segments obtained by intersecting H_1 and H_2 with α must intersect due to Rolo theorem, and their intersection must have non-negative coordinates. By joining the two conditions, one obtains: $w_- = \min_i w_i < 1-\varepsilon_T < \max_i w_i = w_+$. □