# Using energy where it counts: Protecting important messages in the link layer

Andreas Köpke, Holger Karl, Marc Löbbers
koepke@tkn.tu-berlin.de

Berlin, September 16th, 2004

TKN Technical Report TKN-04-014

TKN Technical Reports Series Editor:

Prof. Dr.-Ing. Adam Wolisz

**Abstract**

Saving energy is the most important goal in a sensor network. But short-sighted optimization for energy can lead to sensor networks that can not fulfill their task. Hence, this goal must be balanced with task related goals. One such task related goal is to transmit messages in a sufficiently reliable way. For instance for a monitoring sensor network this means that the messages that arrive at the sink node allow a good overview of the monitored area, while at the same time no energy resources are wasted. In this paper we define a “good overview” as the “informational value” that arrives at the sink node and adapt the reliability of the link layer such that the overall system efficiency is maximized. The system efficiency is defined as the informational value arriving at the sink put into relation with the energy spent by the network to get it there. Our major result is that there exists a rule that describes how to adapt the reliability of the link layer, which can be evaluated by each node using only locally available information. When the reliability of the link layer is adapted according to this rule, the system efficiency can be increased (sometimes by more than 20%) compared to the best performing non-adaptive link layer.

This work has been partially sponsored by the European Commission under the contract IST-2001-34734 – Energy-efficient sensor networks (EYES).

## Contents

## 1 Introduction

Saving energy is an important goal in Wireless Sensor Network (WSN). It is used as the main optimization objective, while other objectives like throughput, delay and reliability are less important. But saving energy alone does not lead to a system that can fulfill a task; because it is very energy-efficient not to do anything at all. As an example, consider the case of a sensor network that monitors a forest for beginning fires or changes in the probability for a fire. To fulfill its task, it can report the temperature and humidity periodically and beginning fires using alarm messages. The alarm messages occur very rarely, but if they occur they are very important and must be transmitted reliably. The periodic reports, on the other hand, consume a large amount of energy, hence their transmission should be optimized to save energy. Furthermore, the periodic reports are less important, because they can be extrapolated either using past sensor readings or readings from other sensors. In this sense, they carry less informational value than the alarm messages. It seems to be a good idea to transmit more important messages more reliably than less important messages, in order to achieve an optimal balance between energy expenditure and reliability. In this paper, we deal with the question how an optimal balance can be found by adapting the reliability of the link layer using the informational value of a message.

The distinction between two classes of messages (periodic reports and alarm messages) is not general enough and does not capture some important cases in sensor networks. In sensor networks, often aggregation mechanisms are used to lower the number of transmitted messages. Such a case is shown in Figure 1.

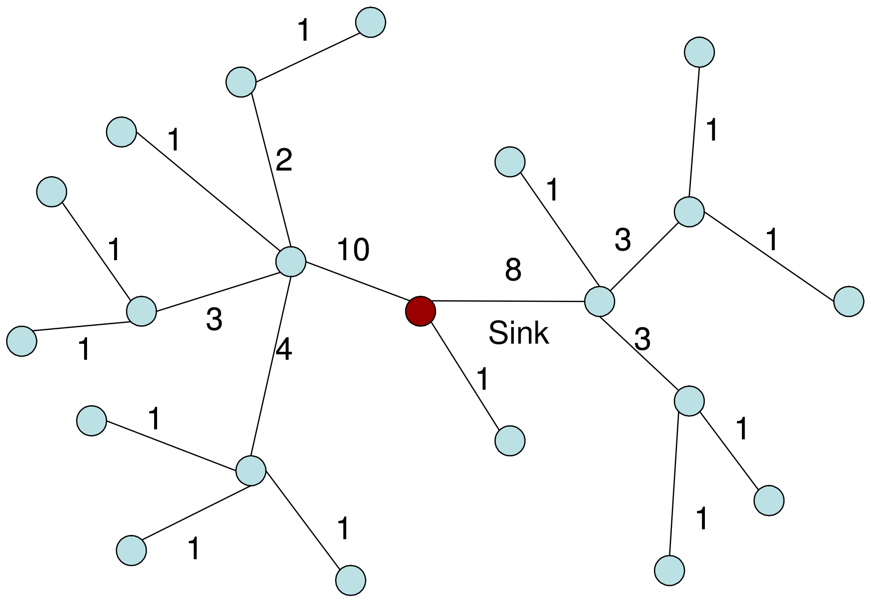

Figure 1: Convergecast tree with number of aggregated sensor readings communicated over each link

In this aggregation example, the nodes report their sensor readings to their parents in the tree. When all children of a node have answered, the messages are combined into a single one and forwarded to the next node. This convergecast is a suitable scheme to transmit the periodic reports in an energy-efficient way. During the convergecast, some messages contain the aggregated readings of more and more nodes. These messages carry more informational value and should therefore be sent more reliably. In this case, it seems to be easy to define a measure for the informational value of a message: just count the number of contributing nodes. While this approach is simple, it is not general enough. We will present other measures for the informational value in Section 3.

Using the (suitably defined) informational value, the link layer of a node chooses a sufficiently reliable way to transmit the message. One way to transmit a message reliably is the transmission using an Automatic Repeat reQuest (ARQ) protocol. An ARQ protocol has at least the overhead of the Acknowledgments (ACKs) plus the retransmissions. The higher the number of possible retransmissions the higher the probability that a message arrives at the parent node, but the higher also the energy spent to send the message.

In the tradeoff between energy expenditure and reliability the overall system efficiency should be maximized. System efficiency is defined as the informational value arriving at the sink node ($v_r$) put into relation with the energy spend to get it there ($\eta_r$). To maximize the system efficiency ($L = v_r/\eta_r$), each node has to decide how reliably a message should be transmitted. Ideally, each node decides locally, without any additional communication. Otherwise the communication overhead may easily exceed the energy savings of a link layer that adapts the reliability of the transmission.

Before we present a local rule to choose the optimal reliable way to transmit a message – the optimal protection of the message by the link layer – in Section 6, we start with a strict mathematical formulation of the optimization problem. The mathematical formulation allows the computation of the system efficiency for a specific way of choosing the protection given the message value. However, the formulation does not easily lead to a rule that can be used by each node to make a decision on the protection it should use. As a major result of this work, we present a rule that every node can evaluate *locally* that approximates the optimal system efficiency very closely.

## 2 Related Work

In this paper, which can be seen as a continuation of [1], we concentrate on the reliable transmission of data from the source to the sink, where each node on the route contributes. More generally, we concentrate on end-to-end reliability with an energy constraint. A similar goal is investigated by STATHOPOULOS and ESTRIN [4]. In their approach, the sink node decides which source node should be asked to retransmit its message. This question is closely related to our approach, however, we concentrate on the question how *each node on the route* can contribute to the reliability, instead of leaving this decision to the sink node. The other direction, from the sink to the sensors is examined by PARK et al. [2]. It remains part of future work to see how their approach can be fitted into a system with messages with different informational value.

If the correlation between the sensor readings is used, a different notion of reliability is more appropriate: the event to sink reliability, introduced in [3]. They assume that sensor readings are by nature correlated and it is sufficient for at least one message to arrive at the sink. From an energy point of view, it is might be better not to send the redundant messages at all, but to compress the messages into a single, high valued one.

At first glance the work done to optimize energy expenditure by building optimal convergecast trees and clusters seems to be related. It is not; aggregation is just used as an example. Optimizing the aggregation structure with some clever clustering or tree building algorithm is not considered in this paper. Furthermore, the local rule that we propose here is completely oblivious to the underlying aggregation structure, therefore we conjecture[1] that an optimal aggregation structure comes as an additional benefit, but has no influence on how reliable a node forwards a message. The specific measure for the informational value that is used as an example has some relation with an aggregation structure, however, our approach is much more general as we will exemplify in the next section.

## 3 Informational Value

The definition of the informational value is not an easy task, since it can not easily be defined without application specific knowledge. In sensor networks, this knowledge is available and the protocol stack can be designed such that this information can be passed to the link layer. This is a unique feature for sensor networks and explains why the “Quality of Service” of conventional networks research is not directly related.

Application specific measures are – when suitably defined – probably the best possible measures for the informational value. Their disadvantage is that someone has to define them carefully, which can be a tedious work. We present some measures for the informational value that are more general, and can be used by more than one application.

### 3.1 Count-based Measure

For the aggregation example, we have already introduced a very simple and readily available measure for the informational content: the count based measure. The informational value of a packet is defined as the number of sensor readings aggregated into a forwarded packet. It is easy to compute and does not need any additional information or support from the network, the convergecast tree is sufficient. In this paper, we use it to compute the system efficiency and to find the optimal system efficiency. But this measure has its shortcomings, especially when the sensors are distributed randomly and some areas are observed by many sensors and some are observed by just a few.

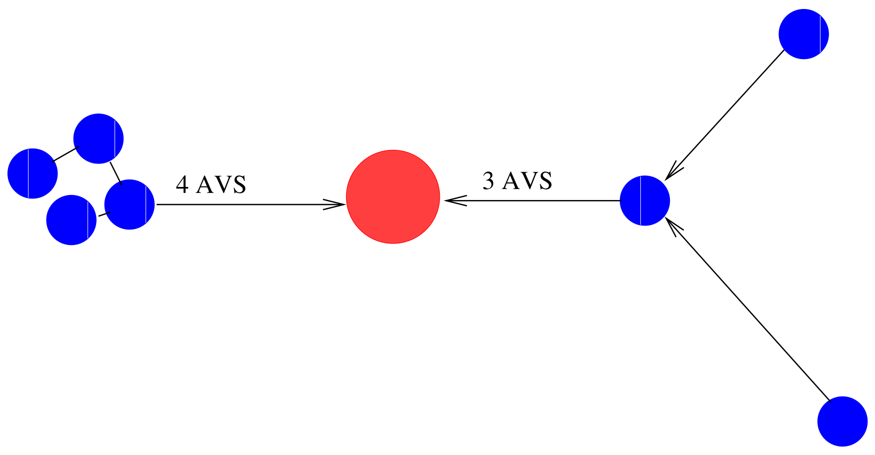

Figure 2: Unequally distributed sensors

## 3.2 Area-based Measures

Figure 2 shows such a case where sensors are unevenly distributed. Here, the left side will contribute more informational value if simply contributing sensors are counted. However, the left side covers a much smaller area than the right side and this should have an influence on the value.

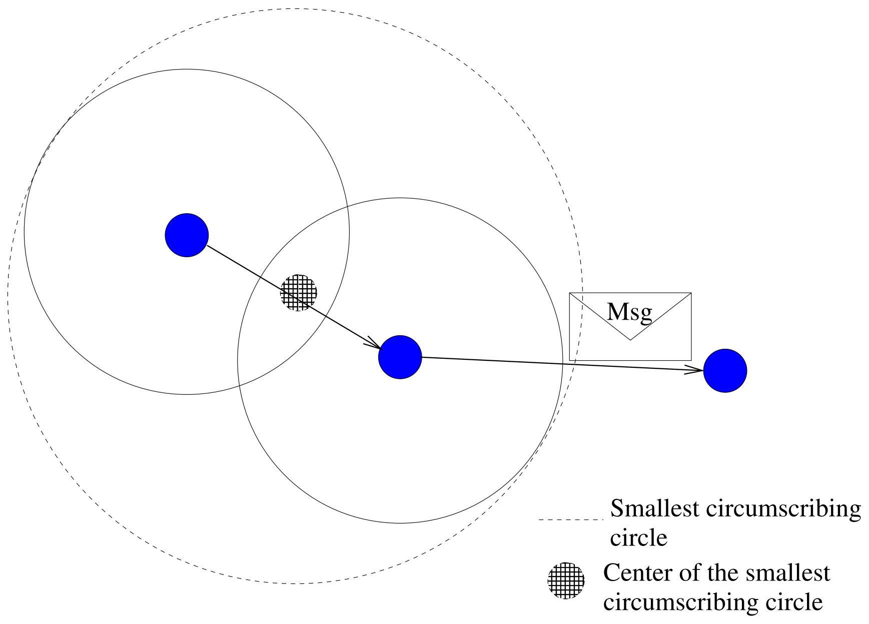
Msg
Smallest circumscribing circle
Center of the smallest circumscribing circle

Figure 3: Circumscribing circle

A possible way to measure the impact of the covered area on the informational value is to assign each sensor an area that it observes and use the area of the circumscribing circle as the measure of the informational value. This is shown in Figure 3. The problem with this measure is that it grows rapidly. Also it does not reflect the

[1]Computation results for other graphs support this claim.

fact how the sensors are distributed in it – if all sensors are in a small area on the left and one is on the far right, it will still assign a large value to the message.

A third possible measure tries to account for that fact and assigns an area to the circle that increases by the area a new sensor contributes. If the sensors are packed in a small area, each sensor will contribute only a small additional area, because its area is already observed by other sensors. Compared to the simple count based measure, the area based measures reflect reality better, but need geographical information.

### 3.3 Entropy based measures

What the area based measures try to compute is the additional information that a new sensor contributes. This additional information is actually the best possible measure and is at the heart of entropy based measures. If a sensor is the only one that observes an area, it contributes much information and lowers the entropy by a large amount for this area. In the forest monitoring example, sensors within a region that has only a few sensors contribute much information, compared to sensors in regions with many sensors. In addition, the sensor that reports a starting fire first provides much information, because this is a surprising and unexpected event. Entropy based measures are able to capture also the latter case. Therefore, we believe such measures are the most appropriate. In future work, we will examine several measures from statistics and coding theory and tailor them to the needs of an adaptive link layer.

## 4 Protection mechanisms

These measures enable an operational definition of the informational value of a message. Using this informational value, a link layer has to decide which protection mechanism it should use. Some conceivable mechanisms include Forwad Error Correction (FEC), transmission power control, data rate adaptation, packet length adaptation, and ARQ protocols.

**FEC:** Forward Error Correction allows to add some redundancy to a packet so that the packet can be correctly received even if it encountered a (limited) number of bit errors during transmission. Different forward error control codes exist so that proper choices for packet size, channel condition, and intended protection are feasible. The trade-off is between reduced packet error rate and longer packet length.

**ARQ:** ARQ protocols enable retransmission of failed packets by sending acknowledgements and detecting missing acknowledgements. The trade-off here is the required overhead for acknowledgements in the correct case against the (compared to FEC) shorter packet length.

**Transmission power adaptation:**
Higher transmission power reduces the packet error rate by improving the signal to noise ratio, but increases energy consumption. This mechanism has to be supported by the radio front end to make sense.

**Data rate adaptation:**

Closely related to the FEC approach, controlling the data rate reduces the time necessary to transmit a given packet for the price of an increased bit error rate. This mechanism has to be supported by the radio front end to make sense.

The goal for an adaptive link layer is to choose these mechanisms (typically in combination) depending on the informational value of a message that is to be forwarded in such a way that the system efficiency is maximized.

# 5 Trading Energy for Value

In order to arrive at an optimal system efficiency, it is important to know the protection mechanism influences the probability that a packet arrives as well as its energy consumption. This enables the computation of the system efficiency, defined as $L = v_r/\eta_r$; the value arriving at the sink node ($v_r$) put into relation with the energy used to get it there ($\eta_r$); and this relation should be maximized. The two variable $v_r$ and $\eta_r$ depend on each other: the value arriving at the root depends on the way how the protection is chosen (the mapping of informational value to protection), because a higher protection increases the probability that a message is transmitted correctly. The chosen protection mechanism influences the consumed energy ($\eta_r$) to get the information to the sink node.

To gain an understanding of the problem, a specific system has to be defined. First of all, a measure for informational value is chosen, here the count based measure is used. Each sensor reading adds “1” to the informational value. This choice is not essential for the problem formulation, in fact any expression for the informational value could be used; the mapping of informational value to protection is independent of the measure. Just for convenience it is assumed that a message that is “worth sending” has at least an informational value of “1” and every measure is scaled in such a way that it keeps this property.

Secondly, a relation between the informational value and the chosen protection is defined: $v \to p(v)$, where $p(v)$ denotes the protection chosen, given a certain informational value $v$. The problem is that nothing about the properties of $p$ is known. To keep it is general as possible, a table is used. The table is defined in pairs $v \to p$. When a message with an informational value of $v$ arrives, the link layer looks into the table, and protects the message against e.g. $p$ bit errors per block. The entries in this table must be chosen such that the maximum system efficiency is achieved.

## 5.1 Computing the value arriving at the sink

Using the definition of the value and the definition of how the protection is chosen, it is possible to compute the system efficiency. The system in Figure 4 is used to derive the computation of the value arriving at the sink node $v_r$. The computation is done recursively, $v_t$ denotes the value that arrives at tree depth $t$.

In WSN messages do not always arrive correctly, but with a certain probability $P$. Messages are lost due to independent bit errors that appear with a certain

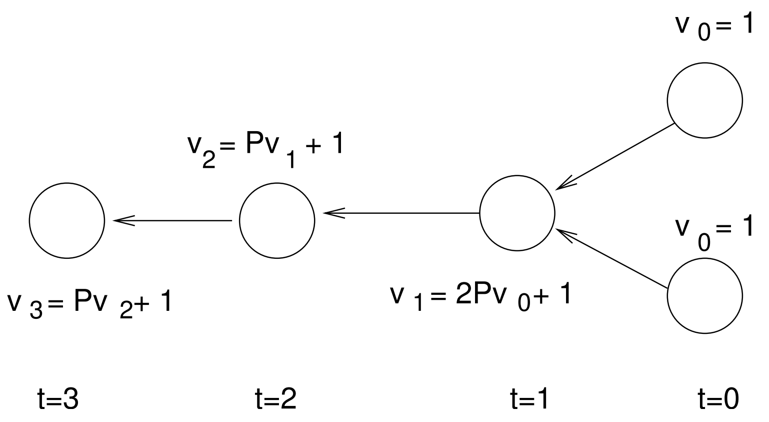

Figure 4: Example problem sketch

probability $b$. This probability is not constant but depends on the protection $p(v)$ of the message transmitted, because messages with higher values are possibly given a higher protection; in which case they arrive with a higher probability.

Thus, the expected informational value of a message that a certain node at tree depth $t+1$ has to forward is:

$$v_{t+1} = \begin{cases} a + \sum_{j=1}^{c} P(v_{t,j}) v_{t,j} & \text{if } c > 0 \\ a & \text{if } c = 0 \end{cases} \tag{1}$$

where $c \in \mathbb{N}_0$ denotes the number of children of the node, and $a$ denotes the value that the node at three depth $t+1$ adds, throughout this paper, a count based measure is used, hence $a = 1$. The forwarded value depends on the probability that a certain value arrives from a subtree that is rooted in the child $j$.

With this value, it is possible to compute the energy spent for the transmission of the message ($\eta_{t+1} = f(p(v_{t+1}))$), but this is specific for each protection mechanism, and discussed in the next sections.

## 5.2 Forward Error Correction

### 5.2.1 Fixed length codes

In this paper, three ways to protect a packet against bit errors are evaluated. The first class are Bose-Chaudhuri-Hocquenghem (BCH) codes with fixed block length. This means that the $l$ information bits are spread across multiple blocks. Each of these blocks of length $n$ can contain $i$ information bits, the $n - i = k$ control bits are necessary to correct a certain amount of bit errors.

For the computation of the system efficiency a BCH code with a block length $n$ of 63 bit is used. The protection $p(v) = e$ is the number of bit errors in a block $e$ that can be corrected. This requires an overhead, namely the number of control bits in a block. The number of control bits $k$ necessary to be able to correct $e$ bit errors were taken from [5, 6] and are shown in Table 1.

If the maximum protection is used, only two information bits per block can be transmitted. For a message that would have fitted into a single 63 bit block when no

| **e** | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 10 | 11 | 13 | 15 | 17 | 20 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **i** | 57 | 51 | 45 | 39 | 36 | 30 | 24 | 18 | 16 | 10 | 7 | 3 | 2 |
| **k** | 6 | 12 | 18 | 24 | 27 | 33 | 39 | 45 | 47 | 53 | 56 | 60 | 61 |

Table 1: Overhead for a 63 bit BCH code

protection is used, 33 blocks or 2079 bit have to be transmitted when the maximum protection is used.

### 5.2.2 Optimal length codes

The fixed block length codes (esp. those with small block sizes like $n = 15\,\mathrm{bit}$) have the advantage that they can be kept in a table, simplifying encoding and decoding to table lookups. But they are not the most efficient way to protect a message against independent bit errors. It is more efficient to make the block so large that it contains all information bits and the control bits in a single block.

The number of control bits $k$ necessary to correct $e$ bit errors in an $n$ bit long block can be approximated using the Hamming distance $d = 2e + 1$ and the Varshamov-Gilbert bound

$$k(d) \geq \log_2 \left( \sum_{j=0}^{d} \binom{n}{j} \right) . \tag{2}$$

The transmitted packet is $n = i + k$ bits long. Also for this protection mechanism the protection $p$ is simply the number of bit errors $e$ that can be corrected in such a packet. The energy spent to transmit the message is related to the length of the packet, and computed in bits.

### 5.2.3 Probability of successful transmission

The energy spent to transmit a message is only one side of the protection mechanism. It also influences the probability that a packet arrives. The probability that a block is successfully transmitted, which means it contains at most the number of bit errors that can be corrected, is given by the Bernoulli distribution:

$$P(X \leq e) = \sum_{j=0}^{e} \binom{n}{j} b^x (1-b)^{n-j} \tag{3}$$

The probability that a packet with $l$ information bits is successfully transmitted using blocks of length $n$ is

$$P(S) = P(X \leq e)^{\lceil l/n \rceil}, \tag{4}$$

since each block in it must be transmitted correctly.

## 5.3 ARQ

A completely different approach to add error protection is the use of an ARQ protocol. Basically, the ARQ protocol sends an ACK for every correctly received packet.

The computation of energy consumption and success probability is more complex, as several cases have to be discerned.

The first case considered here is a successful transmission; the packet and the ACK are received correctly. The probability that a packet consisting of $l$ bits is received without errors is

$$P(A) = (1-b)^l, \tag{5}$$

whereas the probability that an ACK of length $l_a$ is received successfully is

$$P(B) = (1-b)^{l_a}. \tag{6}$$

For a successful transmission it is necessary that at least one of the transmission attempts is successful. If $t$ denotes the maximum number of retransmissions, the probability of a successful transmission $P(S)$ is given by

$$P(S) = P(A)\sum_{i=0}^{t}[1-P(A)]^{li} \tag{7}$$

The expected energy used for one transmission attempt consists of the energy spend for the packet and the energy spend for the ACK. However, the ACK is only send if the packet was received correctly. Hence, the expected energy per transmission attempt $\mathrm{E}(E)$ is

$$\mathrm{E}(E) = P(A)(l+l_a) + [1-P(A)]l \tag{8}$$

The ARQ retries to send a packet when either the packet or the ARQ is lost. This has implications on the expected number of transmissions $\mathrm{E}(T)$:

$$\mathrm{E}(T) = 1 + \sum_{i=1}^{t}[1-P(A)P(B)]^i \tag{9}$$

These formulas allow the computation of the expected value that arrives at the root node and the energy that is on average spend to achieve this: $\eta_{t+1} = \mathrm{E}(T)\mathrm{E}(E)$.

## 5.4 Computing the system efficiency

With the definition of the informational value and the spent energy it is possible to compute the system efficiency:

$$L = \frac{v_r}{\nu} \cdot \frac{(\nu-1)l}{\eta_r} \approx \frac{v_r l}{\eta_r} \tag{10}$$

where $v_r$ is the value arriving at the sink, the root of the spanning tree and $\nu$ is the total value, which is equal to the number of nodes in the tree. Hence $v_r/\nu$ is the value arriving at the root compared with the maximum possible value. The second term $(\nu-1)l/\eta_r$ represents the energy consumption, measured in sent bits ($\eta_r$) compared with the minimal number of bits that must be sent: $(\nu-1)l$ . The minimal energy that has to be spent is equal to the number of edges in the tree $\nu-1$ times the number of information bits $l$ that must be transmitted.

The formulas presented in the preceding sections are difficult to evaluate symbolically, hence they were evaluated numerically using a network with 400 nodes in a unit disc graph, where each node had 14 neighbors on average. The spanning tree was built using Dijsktra's shortest path algorithm[2]. To compute the overall system efficiency, the system efficiencies for every of the 400 possible sink nodes were averaged.

In this setup, we tried to find the optimal system efficiency for a number of Bit Error Rates (BERs) using different ways to map the informational value to protection ($v \to p(v)$). Finding the optimal solution is difficult, because the formulation is still quite open and allows many solutions. First of all, it is unknown how this mapping should look like. In order to gain an understanding of the problem, the mapping was not expressed in a functional relationship like e.g. $v \to p(v) = \alpha + \beta v^{\gamma}$, but in a table. Using the value, the link layer looks up the protection to use in this table. Still, this does not lower the complexity of the optimization problem, because it is for instance unknown how many entries the mapping should have. Is one enough? Or should it contain 10, 20 or more entries?

Fortunately, the entries in the mapping table are independent. If there is only one entry, it is sufficient to find the optimal solution for this entry. For a count based measure of the informational value, the necessary number of computations is smaller than the number of nodes in the system. In next step, this entry is kept fixed (say $1 \to 1$ that is if the informational value is greater then 1, protect the packet against a single bit error per block) and the next entry is tried. If the optimum is found (say $2 \to 3$), this entry is also kept and the next entry is tried. This way, the optimal mapping can be found in an iterative fashion.

## 6 Choosing the right protection

In a deployed sensor network it is not possible to find the optimal mapping of informational value to protection in such an iterative way. It imposes a prohibitively large communication overhead, as not only many mappings have to be tried but also each mapping has to be kept for a certain time to gain an understanding of its performance.

In order to adapt the reliability of the link layer given the informational value of a message, it is necessary that each node on the route can choose the protection with locally available information. We propose the following rule, which allows a very good approximation of the globally optimal mapping: increase the protection as long as the increase in value outweighs the relative increase in energy, Listing 1 shows the rule in pseudo-code.

This local rule allows a very close approximation of the maximum system efficiency and has several interesting features. First of all, it is completely independent of the protection mechanism used (BCH, ARQ, transmission power adaptation, data rate adaptation, etc.) as long as each mechanism can be expressed in increased energy expenditure. Secondly, it is completely oblivious of the underlying network topology. This suggests that it is useful under a wide range of conditions and thus provides a valuable starting point for anyone searching the right tradeoff between energy and the value that arrives at the root node.

[2]For comparison the computations were also done for random graphs, where the spanning tree was built with Prim's algorithm. The results are similar and therefore not shown here.

Listing 1: Pseudo-code of local rule

```
current protection = no protection;
oldPs = P(S) using current
protection newPs = P(S) using increased protection;
oldEnergy = energy consumption using current protection;
newEnergy = energy consumption using increased protection;

while((newPs−oldPs)∗value >=
      (newEnergy − oldEnergy)/newEnergy)
{
  current protection = increased protection;
  oldPs = newPs;
  oldEnergy = newEnergy;
  increase protection;
  compute values for newPs and newEnergy
  for this new, increased protection;
}
transmit using current protection;
```

## 7 Results

A link layer that adapts the reliability of transmission to the informational value of a message implies an increased complexity in the protocol stack. This complexity has to pay off, and in this section we present the maximum achievable system efficiencies for each of the three approaches (globally optimum, local rule and fixed protection) for the three protection mechanisms (FEC with fixed block length, FEC with optimal block length and an ARQ protocol) for different BERs.

Table 2: Value to protection mappings

| Algo. | Size | BER | G. Map. | L. Map. | G. Opt. | L. Opt. | F. Opt |
|---|---|---|---|---|---|---|---|
| VG | 300 | 0.0005 | $1 \to 1$<br>$4 \to 2$<br>$47 \to 3$ | $1 \to 1$<br>$4 \to 2$<br>$44 \to 3$ | 0.91 | 0.91 | 0.90 |
| VG | 300 | 0.001 | $1 \to 1$<br>$2 \to 2$<br>$7 \to 3$<br>$86 \to 4$ | $1 \to 1$<br>$2 \to 2$<br>$7 \to 3$<br>$81 \to 4$ | 0.89 | 0.89 | 0.83 |
| VG | 300 | 0.005 | $1 \to 4$<br>$2 \to 5$<br>$4 \to 6$<br>$12 \to 7$<br>$41 \to 8$ | $1 \to 4$<br>$2 \to 5$<br>$4 \to 6$<br>$11 \to 7$<br>$38 \to 8$<br>$145 \to 9$ | 0.80 | 0.80 | 0.76 |
| BCH | 300 | 0.0005 | $2 \to 2$ | $2 \to 2$ | **0.81** | **0.81** | **0.63** |

*continued on next page*

Table 2: Value to protection mappings

| **Algo.** | **Size** | **BER** | **G. Map.** | **L. Map.** | **G. Opt.** | **L. Opt.** | **F. Opt** |
|---|---|---|---|---|---|---|---|
| BCH | 300 | 0.001 | $1 \rightarrow 2$ | $1 \rightarrow 2$ | 0.80 | 0.80 | 0.80 |
| BCH | 300 | 0.005 | $1 \rightarrow 2$ | $1 \rightarrow 2$ | 0.76 | 0.76 | 0.74 |
| | | | $8 \rightarrow 3$ | $7 \rightarrow 3$ | | | |
| | | | $85 \rightarrow 4$ | $66 \rightarrow 4$ | | | |
| ARQ | 300 | 0.0005 | $2 \rightarrow 2$ | $2 \rightarrow 3$ | **0.70** | **0.70** | **0.55** |
| | | | $3 \rightarrow 4$ | $3 \rightarrow 5$ | | | |
| | | | $4 \rightarrow 5$ | $4 \rightarrow 6$ | | | |
| | | | $5 \rightarrow 7$ | $5 \rightarrow 7$ | | | |
| | | | $6 \rightarrow 8$ | $6 \rightarrow 8$ | | | |
| ARQ | 300 | 0.001 | $2 \rightarrow 2$ | $2 \rightarrow 3$ | **0.61** | **0.61** | **0.39** |
| | | | $3 \rightarrow 4$ | $3 \rightarrow 5$ | | | |
| | | | $4 \rightarrow 6$ | $4 \rightarrow 7$ | | | |
| | | | $5 \rightarrow 7$ | $5 \rightarrow 8$ | | | |
| ARQ | 300 | 0.005 | $3 \rightarrow 11$ | $3 \rightarrow 15$ | 0.17 | 0.17 | 0.06 |
| | | | $4 \rightarrow 15$ | $4 \rightarrow 19$ | | | |
| | | | $5 \rightarrow 18$ | $5 \rightarrow 22$ | | | |
| | | | $6 \rightarrow 21$ | $6 \rightarrow 24$ | | | |

The results are shown in Table 2. The column "Algo." denotes the protection algorithm used, VG stands for an FEC with optimal block lengths, BCH for a BCH code with a block length of 63 bit. For ARQ an ACK of 80 bits was used. The column "size" denotes the packet size in bits. The next column contains the BER.

The next column (G. Map.) contains the mapping of values to the number of bit errors that the BCH codes should be able to correct, or the number of *re*transmissions for an ARQ protocol. This mapping was obtained in the iterative fashion described in section 5.4.

Column L. Map. is the mapping obtained with the local rule presented in the previous section. A comparison with the global optimum shows that the mappings differ only marginally – this can also be seen by comparing the values of $L$ for these mappings, presented in columns G. Opt and L. Opt. The maximum possible value is $400/399 \approx 1$. The last column shows the value of $L$ if the link layer with the best fixed protection is used. The best fixed protection is also contained in the table, using the independence of table entries: if the mapping contains an entry $1 \rightarrow p$ this is the best fixed protection, if the table does not contain such an entry, the best fixed protection scheme is to use no protection at all. We decided to compare against the best possible fixed protection, assuming that the designer of a link layer chooses the reliability of the link layer carefully with the application in mind. Other choices for the fixed protection are always somewhat arbitrary, as it is possible to choose a scheme that yields a system efficiency close to zero.

The table shows that the optimal mapping and the mapping computed with the local rule yield practically the same performance. It also shows that an adaptive link layer can be nearly twice as good compared to one with fixed protection. If an adaptive link layer can not be implemented, the application of the local rule leads to link layers with fixed protection that perform usually nearly as good as the adaptive ones.

Another interesting fact is that the ARQ protocol should not be used if a packet has only a value of one. Put differently, if ARQ is used as a mechanism, the network should never retransmit messages from leaf nodes.

The results can be summarized in two points: if possible, use a FEC with long blocks and choose a fixed protection with the help of the local rule. Although the fixed protection with block codes performs poorly under some scenarios, an adaptive scheme may not be worth the effort. The reason for the often negligible difference between the fixed protection scheme and the adaptive ones is the granularity how the protection mechanism can be adjusted. The optimal block length FEC allows the most granular increase. Protecting the packet against on more bit error increases the overall packet length slightly. But even this small increase is often too large and a considerable increase in the value (e.g. from 4 to 47) is needed before it pays off. The effect becomes even more pronounced when a less granular scheme like the BCH code with 63 bit block size is used. Here, choosing a fixed basic protection with a block code is the optimal strategy for many BERs.

The second point is that an ARQ protocol performs considerable worse than the other schemes. However, it is often built into the link layer, for instance in many Carrier Sense Multiple Access (CSMA) type Medium Access Control (MAC) protocols it is used to detect collisions of data packets; ARQ protocols are also necessary for hop-by-hop flow control, which also explains why they are used so often in link layers. In addition, an ARQ protocol should perform better when the bit errors are not independent but appear in bursts. It remains part of future work to evaluate performance for transmission channels with bursty bit errors. It seems to be interesting to make a more in depth simulation for ARQ protocols because the results in Table 2 suggest that it is possible to find an adaptive ARQ protocol that works reasonably well under a wide range of conditions.

## 8 Conclusion and Future Work

The evaluation of different ways to adapt the reliability of a link layer to the informational value of a message lead to a number of interesting insights. Using the system efficiency it is possible to define an optimal balance between energy expenditure and the informational value that arrives at the sink node. The local rule presented in this paper shows that each node can decide how reliable it should transmit a message just based on the informational value of the message and some channel information. The adaptation of the reliability of a link layer is nonetheless challenging, because the available protection mechanisms do not allow a sufficiently fine grained control of the reliability. It is therefore often optimal to choose a fixed protection scheme with the local rule at design time.

A slightly different conclusion has to be drawn for ARQ protocols. ARQ protocols are used very often in link layers, esp. when a CSMA type MAC is used. In addition, they allow hop-by-hop flow control and thus provide an added value. Although they perform poorly under the conditions examined in this paper, they should perform better for more realistic assumptions for the bit error behavior. Hence, the performance of adaptive ARQ protocols should be studied in depth for different channel assumptions as well as different ways to measure the informational content, at the very least the ARQ protocols should be analyzed for other aggregation structures and other channel conditions.

## References

[1] H. Karl, M. Löbbers, and T. Nieberg. A Data Aggregation Framework for Wireless Sensor Networks. In *Proc. Dutch Technology Foundation ProRISC Workshop on Circuits, Systems and Signal Processing*, November 2003.

[2] Seung-Jong Park, Ramanuja Vedantham, Raghupathy Sivakumar, and Ian F. Akyildiz. A scalable approach for reliable downstream data delivery in wireless sensor networks. In *Proceedings of the 5th ACM international symposium on Mobile ad hoc networking and computing*, pages 78–89. ACM Press, 2004.

[3] Yogesh Sankarasubramaniam, B. Akan, and Ian F. Akyildiz. ESRT: event-to-sink reliable transport in wireless sensor networks. In *Proceedings of the 4th ACM international symposium on Mobile ad hoc networking & computing*, pages 177–188. ACM Press, 2003.

[4] Thanos Stathopoulos and Deborah Estrin. An Information-Driven Reliability Mechanism for Wireless Sensor Networks. Technical Report 16, Center for embedded networked sensors (CENS), June 2003.

[5] J. P. Stenbit. Table of generators for Bose-Chaudhuri codes. *IEEE Trans. Inf. Theory*, IT(10):390–391, 1964.

[6] Claus Wilhelm. *Datenübertragung*. Militärverlag der DDR, Berlin, 1976.

## A Additional Graphs

The evaluations were made for a range of parameters. The BER, the packet size and the algorithm were varied to gain an uderstanding of the process.

### A.1 Unit disc graph, Dijkstra tree, compare mappings

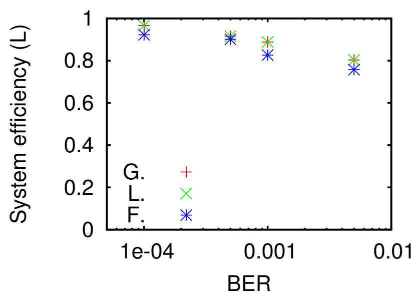

Figure 5: Varshamov-Gilbert, 300 bits “payload” per packet

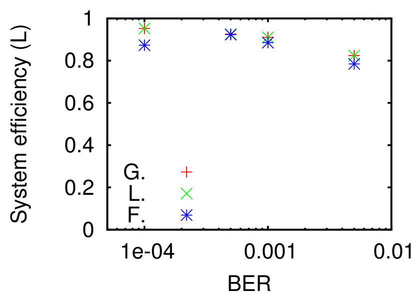

Figure 6: Varshamov-Gilbert, 500 bits “payload” per packet

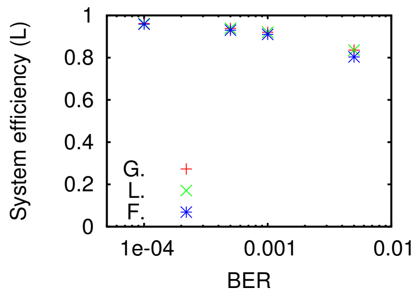

Figure 7: Varshamov-Gilbert, 700 bits “payload” per packet

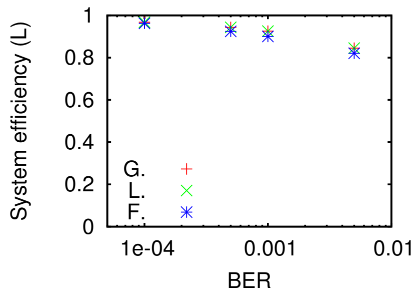

Figure 8: Varshamov-Gilbert, 900 bits “payload” per packet

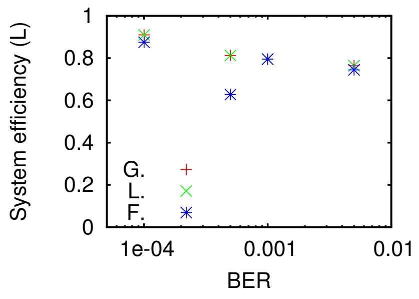

Figure 9: BCH 63 bit long blocks, 300 bits “payload” per packet

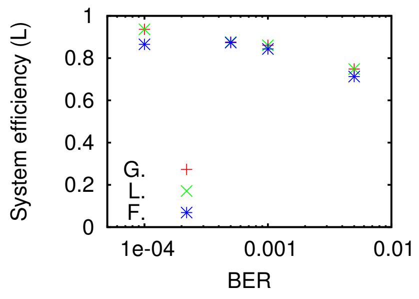

Figure 10: BCH 63 bit long blocks, 500 bits “payload” per packet

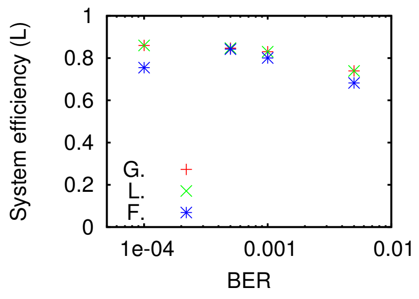

Figure 11: BCH 63 bit long blocks, 700 bits “payload” per packet

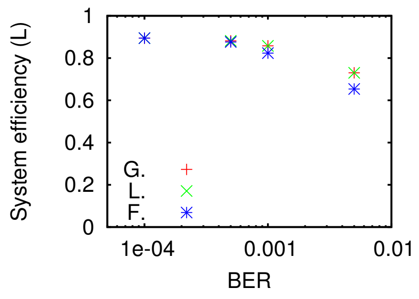

Figure 12: BCH 63 bit long blocks, 900 bits “payload” per packet

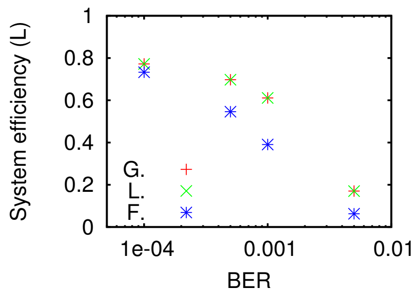

Figure 13: ARQ 80 bit long ACKs, 300 bits “payload” per packet

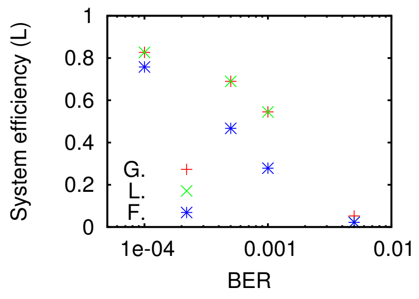

Figure 14: ARQ 80 bit long ACKs, 500 bits “payload” per packet

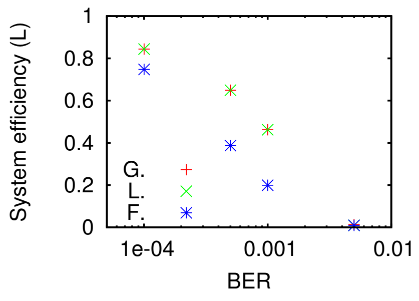

Figure 15: ARQ 80 bit long ACKs, 700 bits “payload” per packet

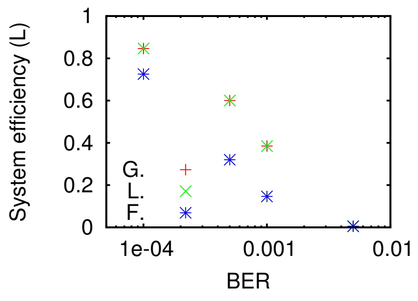

Figure 16: ARQ 80 bit long ACKs, 900 bits “payload” per packet

## A.2 Random graph, Prim's algorithm tree, compare mappings

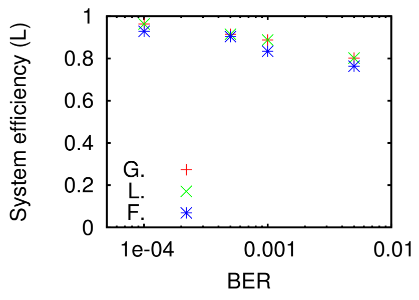

Figure 17: Varshamov-Gilbert, 300 bits “payload” per packet

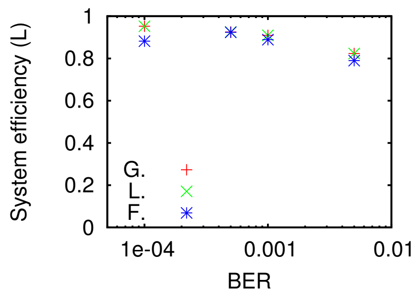

Figure 18: Varshamov-Gilbert, 500 bits “payload” per packet

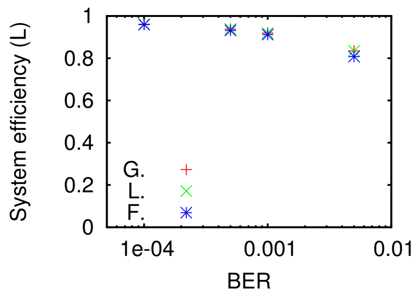

Figure 19: Varshamov-Gilbert, 700 bits “payload” per packet

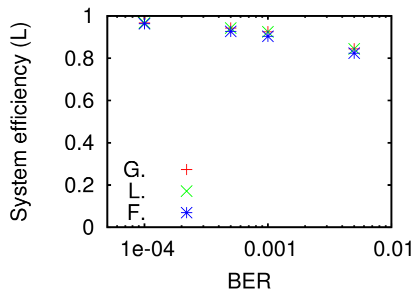

Figure 20: Varshamov-Gilbert, 900 bits “payload” per packet

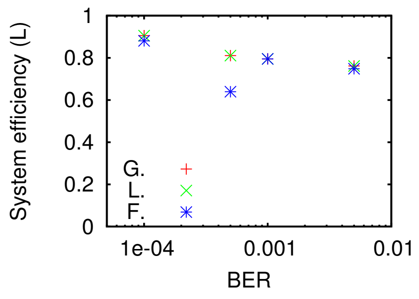

Figure 21: BCH 63 bit long blocks, 300 bits “payload” per packet

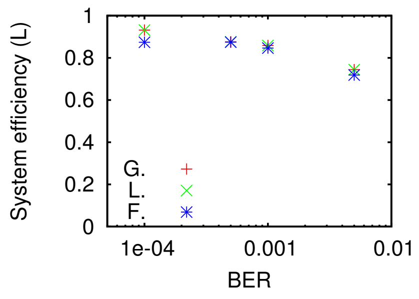

Figure 22: BCH 63 bit long blocks, 500 bits “payload” per packet

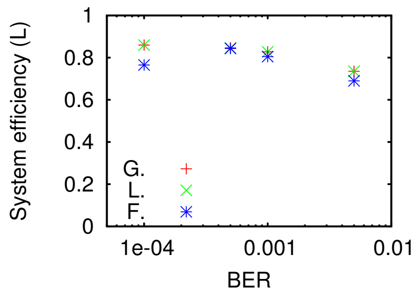

Figure 23: BCH 63 bit long blocks, 700 bits “payload” per packet

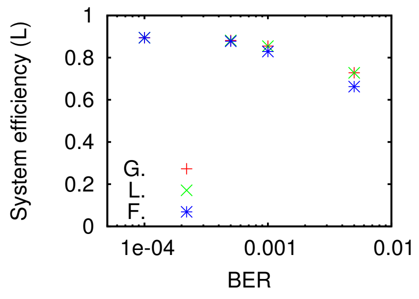

Figure 24: BCH 63 bit long blocks, 900 bits “payload” per packet

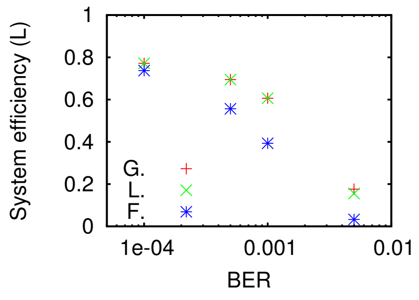

Figure 25: ARQ 80 bit long ACKs, 300 bits “payload” per packet

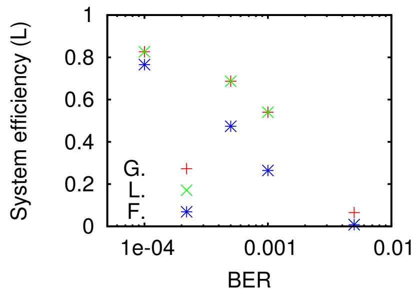

Figure 26: ARQ 80 bit long ACKs, 500 bits “payload” per packet

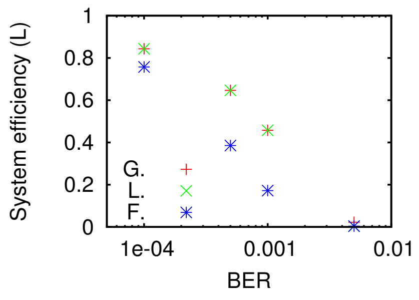

Figure 27: ARQ 80 bit long ACKs, 700 bits “payload” per packet

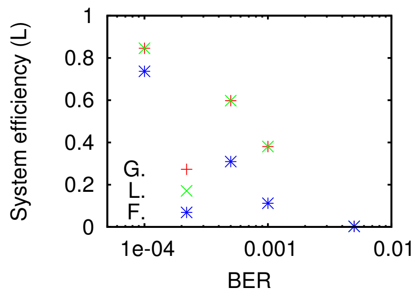

Figure 28: ARQ 80 bit long ACKs, 900 bits “payload” per packet

## A.3 Unit disc graph, Dijkstra tree, compare protection algorithms

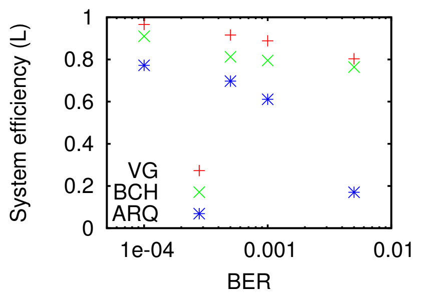

Figure 29: Optimal system efficiency, 300 bits “payload” per packet

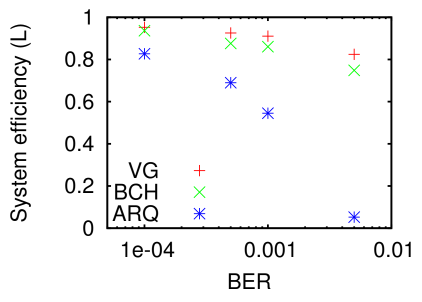

Figure 30: Optimal system efficiency, 500 bits “payload” per packet

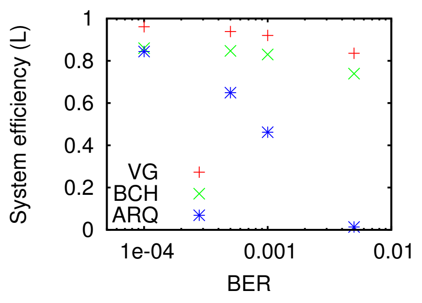

Figure 31: Optimal system efficiency, 700 bits “payload” per packet

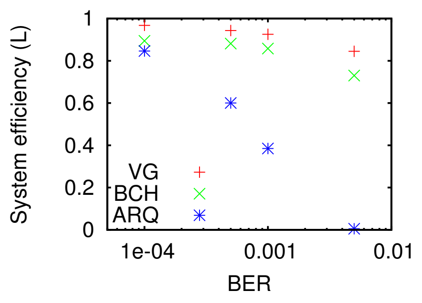

Figure 32: Optimal system efficiency, 900 bits “payload” per packet

## A.4 Random graph, Prim's algorithm, compare protection algorithms

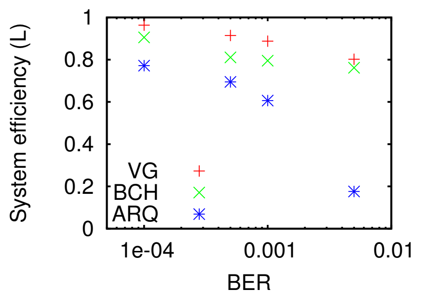

Figure 33: Optimal system efficiency, 300 bits “payload” per packet

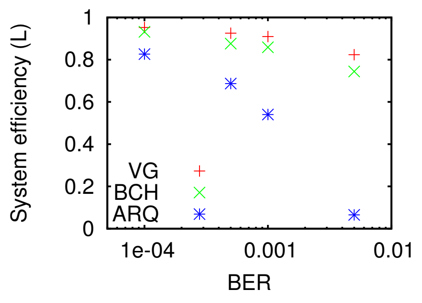

Figure 34: Optimal system efficiency, 500 bits “payload” per packet

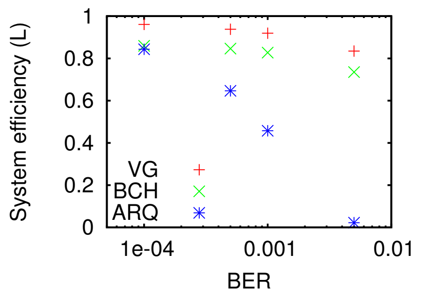

Figure 35: Optimal system efficiency, 700 bits “payload” per packet

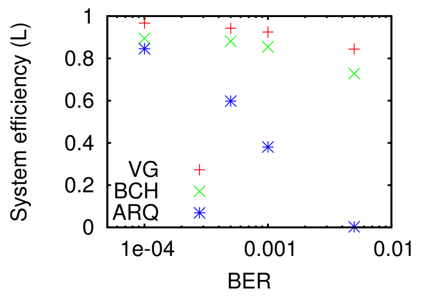

Figure 36: Optimal system efficiency, 900 bits “payload” per packet

# B Complete data as a table

## B.1 Unit disk graph, tree build with Dijsktra

Table 3: Optimal value to protection mappings

| **Algo.** | **Size** | **BER** | **Opt. Map.** | **Local Map.** | **G. Opt.** | **L. Opt.** | **F. Opt** |
|---|---|---|---|---|---|---|---|
| VG | 300 | 0.0001 | 3 $\rightarrow$ 1 | 3 $\rightarrow$ 1<br>83 $\rightarrow$ 2 | 0.96595 | 0.96600 | 0.92220 |
| VG | 300 | 0.0005 | 1 $\rightarrow$ 1<br>4 $\rightarrow$ 2<br>47 $\rightarrow$ 3 | 1 $\rightarrow$ 1<br>4 $\rightarrow$ 2<br>44 $\rightarrow$ 3 | 0.91572 | 0.91572 | 0.90096 |
| VG | 300 | 0.001 | 1 $\rightarrow$ 1<br>2 $\rightarrow$ 2<br>7 $\rightarrow$ 3<br>86 $\rightarrow$ 4 | 1 $\rightarrow$ 1<br>2 $\rightarrow$ 2<br>7 $\rightarrow$ 3<br>81 $\rightarrow$ 4 | 0.88845 | 0.88845 | 0.82649 |
| VG | 300 | 0.005 | 1 $\rightarrow$ 4<br>2 $\rightarrow$ 5<br>4 $\rightarrow$ 6<br>12 $\rightarrow$ 7<br>41 $\rightarrow$ 8 | 1 $\rightarrow$ 4<br>2 $\rightarrow$ 5<br>4 $\rightarrow$ 6<br>11 $\rightarrow$ 7<br>38 $\rightarrow$ 8<br>145 $\rightarrow$ 9 | 0.80302 | 0.80302 | 0.75741 |
| VG | 500 | 0.0001 | 2 $\rightarrow$ 1<br>22 $\rightarrow$ 2 | 2 $\rightarrow$ 1<br>20 $\rightarrow$ 2 | 0.95229 | 0.95229 | 0.87308 |
| VG | 500 | 0.0005 | 1 $\rightarrow$ 2<br>11 $\rightarrow$ 3 | 1 $\rightarrow$ 2<br>11 $\rightarrow$ 3<br>117 $\rightarrow$ 4 | 0.92543 | 0.92544 | 0.92305 |
| VG | 500 | 0.001 | 1 $\rightarrow$ 2<br>2 $\rightarrow$ 3<br>11 $\rightarrow$ 4<br>83 $\rightarrow$ 5 | 1 $\rightarrow$ 2<br>2 $\rightarrow$ 3<br>10 $\rightarrow$ 4<br>79 $\rightarrow$ 5 | 0.91040 | 0.91040 | 0.88531 |
| VG | 500 | 0.005 | 1 $\rightarrow$ 6<br>2 $\rightarrow$ 7<br>3 $\rightarrow$ 8<br>8 $\rightarrow$ 9<br>18 $\rightarrow$ 10<br>56 $\rightarrow$ 11 | 1 $\rightarrow$ 6<br>2 $\rightarrow$ 7<br>3 $\rightarrow$ 8<br>8 $\rightarrow$ 9<br>17 $\rightarrow$ 10<br>53 $\rightarrow$ 11<br>172 $\rightarrow$ 12 | 0.82440 | 0.82440 | 0.78458 |
| VG | 700 | 0.0001 | 1 $\rightarrow$ 1<br>8 $\rightarrow$ 2 | 1 $\rightarrow$ 1<br>8 $\rightarrow$ 2<br>297 $\rightarrow$ 3 | 0.96125 | 0.96125 | 0.95847 |

*continued on next page*

Table 3: Optimal value to protection mappings

| **Algo.** | **Size** | **BER** | **Opt. Map.** | **Local Map.** | **G. Opt.** | **L. Opt.** | **F. Opt** |
|---|---|---|---|---|---|---|---|
| VG | 700 | 0.0005 | 1 $\rightarrow$ 2 | 1 $\rightarrow$ 2 | 0.93861 | 0.93861 | 0.93009 |
| | | | 4 $\rightarrow$ 3 | 4 $\rightarrow$ 3 | | | |
| | | | 34 $\rightarrow$ 4 | 32 $\rightarrow$ 4 | | | |
| | | | | 328 $\rightarrow$ 5 | | | |
| VG | 700 | 0.001 | 1 $\rightarrow$ 3 | 1 $\rightarrow$ 3 | 0.91994 | 0.91995 | 0.91026 |
| | | | 3 $\rightarrow$ 4 | 3 $\rightarrow$ 4 | | | |
| | | | 16 $\rightarrow$ 5 | 16 $\rightarrow$ 5 | | | |
| | | | | 108 $\rightarrow$ 6 | | | |
| VG | 700 | 0.005 | 1 $\rightarrow$ 8 | 1 $\rightarrow$ 8 | 0.83553 | 0.83553 | 0.80340 |
| | | | 2 $\rightarrow$ 10 | 2 $\rightarrow$ 10 | | | |
| | | | 5 $\rightarrow$ 11 | 5 $\rightarrow$ 11 | | | |
| | | | 16 $\rightarrow$ 12 | 15 $\rightarrow$ 12 | | | |
| | | | 36 $\rightarrow$ 13 | 34 $\rightarrow$ 13 | | | |
| | | | 102 $\rightarrow$ 14 | 97 $\rightarrow$ 14 | | | |
| | | | | 294 $\rightarrow$ 15 | | | |
| VG | 900 | 0.0001 | 1 $\rightarrow$ 1 | 1 $\rightarrow$ 1 | 0.96765 | 0.96766 | 0.96232 |
| | | | 5 $\rightarrow$ 2 | 5 $\rightarrow$ 2 | | | |
| | | | | 116 $\rightarrow$ 3 | | | |
| VG | 900 | 0.0005 | 1 $\rightarrow$ 2 | 1 $\rightarrow$ 2 | 0.94327 | 0.94328 | 0.92423 |
| | | | 2 $\rightarrow$ 3 | 2 $\rightarrow$ 3 | | | |
| | | | 12 $\rightarrow$ 4 | 11 $\rightarrow$ 4 | | | |
| | | | | 105 $\rightarrow$ 5 | | | |
| VG | 900 | 0.001 | 1 $\rightarrow$ 3 | 1 $\rightarrow$ 3 | 0.92535 | 0.92535 | 0.90019 |
| | | | 2 $\rightarrow$ 4 | 2 $\rightarrow$ 4 | | | |
| | | | 6 $\rightarrow$ 5 | 6 $\rightarrow$ 5 | | | |
| | | | 28 $\rightarrow$ 6 | 26 $\rightarrow$ 6 | | | |
| | | | | 199 $\rightarrow$ 7 | | | |
| VG | 900 | 0.005 | 1 $\rightarrow$ 10 | 1 $\rightarrow$ 10 | 0.84497 | 0.84497 | 0.82071 |
| | | | 2 $\rightarrow$ 11 | 2 $\rightarrow$ 11 | | | |
| | | | 3 $\rightarrow$ 12 | 3 $\rightarrow$ 12 | | | |
| | | | 5 $\rightarrow$ 13 | 5 $\rightarrow$ 13 | | | |
| | | | 14 $\rightarrow$ 14 | 14 $\rightarrow$ 14 | | | |
| | | | 29 $\rightarrow$ 15 | 27 $\rightarrow$ 15 | | | |
| | | | 76 $\rightarrow$ 16 | 72 $\rightarrow$ 16 | | | |
| | | | | 200 $\rightarrow$ 17 | | | |
| BCH | 300 | 0.0001 | 7 $\rightarrow$ 2 | 6 $\rightarrow$ 2 | 0.90977 | 0.90968 | 0.87466 |
| BCH | 300 | 0.0005 | 2 $\rightarrow$ 2 | 2 $\rightarrow$ 2 | 0.81246 | 0.81246 | 0.62732 |
| BCH | 300 | 0.001 | 1 $\rightarrow$ 2 | 1 $\rightarrow$ 2 | 0.79513 | 0.79513 | 0.79513 |

*continued on next page*

Table 3: Optimal value to protection mappings

| **Algo.** | **Size** | **BER** | **Opt. Map.** | **Local Map.** | **G. Opt.** | **L. Opt.** | **F. Opt** |
|---|---|---|---|---|---|---|---|
| BCH | 300 | 0.005 | $1 \rightarrow 2$<br>$8 \rightarrow 3$<br>$85 \rightarrow 4$ | $1 \rightarrow 2$<br>$7 \rightarrow 3$<br>$66 \rightarrow 4$ | 0.76393 | 0.76383 | 0.74442 |
| BCH | 500 | 0.0001 | $3 \rightarrow 1$ | $3 \rightarrow 1$ | 0.93611 | 0.93611 | 0.86521 |
| BCH | 500 | 0.0005 | $1 \rightarrow 1$<br>$27 \rightarrow 2$ | $1 \rightarrow 1$<br>$24 \rightarrow 2$ | 0.87571 | 0.87571 | 0.87342 |
| BCH | 500 | 0.001 | $1 \rightarrow 1$<br>$7 \rightarrow 2$ | $1 \rightarrow 1$<br>$7 \rightarrow 2$ | 0.86064 | 0.86064 | 0.84327 |
| BCH | 500 | 0.005 | $1 \rightarrow 2$<br>$6 \rightarrow 3$<br>$29 \rightarrow 4$ | $1 \rightarrow 2$<br>$5 \rightarrow 3$<br>$24 \rightarrow 4$<br>$336 \rightarrow 5$ | 0.74817 | 0.74797 | 0.71254 |
| BCH | 700 | 0.0001 | $2 \rightarrow 1$ | $2 \rightarrow 1$<br>$284 \rightarrow 2$ | 0.85978 | 0.85978 | 0.75451 |
| BCH | 700 | 0.0005 | $1 \rightarrow 1$<br>$13 \rightarrow 2$ | $1 \rightarrow 1$<br>$12 \rightarrow 2$ | 0.84703 | 0.84702 | 0.84202 |
| BCH | 700 | 0.001 | $1 \rightarrow 1$<br>$4 \rightarrow 2$ | $1 \rightarrow 1$<br>$4 \rightarrow 2$<br>$240 \rightarrow 3$ | 0.82980 | 0.82980 | 0.80048 |
| BCH | 700 | 0.005 | $1 \rightarrow 2$<br>$3 \rightarrow 3$<br>$32 \rightarrow 4$ | $1 \rightarrow 2$<br>$3 \rightarrow 3$<br>$26 \rightarrow 4$<br>$341 \rightarrow 5$ | 0.73909 | 0.73905 | 0.68234 |
| BCH | 900 | 0.0001 | $1 \rightarrow 1$ | $1 \rightarrow 1$<br>$358 \rightarrow 2$ | 0.89431 | 0.89431 | 0.89431 |
| BCH | 900 | 0.0005 | $1 \rightarrow 1$<br>$17 \rightarrow 2$ | $1 \rightarrow 1$<br>$15 \rightarrow 2$ | 0.88161 | 0.88158 | 0.87608 |
| BCH | 900 | 0.001 | $1 \rightarrow 1$<br>$5 \rightarrow 2$ | $1 \rightarrow 1$<br>$4 \rightarrow 2$<br>$149 \rightarrow 3$ | 0.85796 | 0.85785 | 0.82332 |
| BCH | 900 | 0.005 | $1 \rightarrow 2$<br>$2 \rightarrow 3$<br>$38 \rightarrow 4$ | $1 \rightarrow 2$<br>$2 \rightarrow 3$<br>$31 \rightarrow 4$<br>$102 \rightarrow 5$ | 0.73008 | 0.73002 | 0.65372 |

*continued on next page*

Table 3: Optimal value to protection mappings

| **Algo.** | **Size** | **BER** | **Opt. Map.** | **Local Map.** | **G. Opt.** | **L. Opt.** | **F. Opt** |
|---|---|---|---|---|---|---|---|
| ARQ | 300 | 0.0001 | 2 → 2<br>3 → 4<br>6 → 5 | 2 → 3<br>3 → 4<br>4 → 5<br>5 → 6<br>6 → 7<br>7 → 8<br>9 → 9<br>10 → 10 | 0.77199 | 0.77199 | 0.73261 |
| ARQ | 300 | 0.0005 | 2 → 2<br>3 → 4<br>4 → 5<br>5 → 7<br>6 → 8<br>8 → 10 | 2 → 3<br>3 → 5<br>4 → 6<br>5 → 7<br>6 → 8<br>7 → 9<br>9 → 10<br>11 → 11<br>13 → 12<br>17 → 13<br>21 → 14 | 0.69731 | 0.69731 | 0.54599 |
| ARQ | 300 | 0.001 | 2 → 2<br>3 → 4<br>4 → 6<br>5 → 7<br>6 → 8<br>7 → 9<br>8 → 10<br>9 → 11<br>10 → 12<br>13 → 14 | 2 → 3<br>3 → 5<br>4 → 7<br>5 → 8<br>6 → 9<br>7 → 10<br>9 → 11<br>11 → 12<br>13 → 13<br>15 → 14<br>19 → 15 | 0.61108 | 0.61107 | 0.39058 |
| ARQ | 300 | 0.005 | 3 → 11<br>4 → 15<br>5 → 18<br>6 → 21<br>7 → 23<br>8 → 24<br>9 → 25<br>10 → 27<br>11 → 28<br>12 → 29<br>13 → 30 | 3 → 15<br>4 → 19<br>5 → 22<br>6 → 24<br>7 → 25<br>8 → 27<br>9 → 28<br>10 → 29<br>11 → 30<br>12 → 31<br>13 → 32 | 0.17035 | 0.17015 | 0.062809 |

*continued on next page*

Table 3: Optimal value to protection mappings

| Algo. | Size | BER | Opt. Map. | Local Map. | G. Opt. | L. Opt. | F. Opt |
|---|---|---|---|---|---|---|---|
| ARQ | 500 | 0.0001 | $2 \rightarrow 5$ | $2 \rightarrow 4$ | 0.82673 | 0.82673 | 0.75775 |
| | | | $3 \rightarrow 6$ | $3 \rightarrow 7$ | | | |
| | | | | $4 \rightarrow 9$ | | | |
| | | | | $5 \rightarrow 12$ | | | |
| ARQ | 500 | 0.0005 | $2 \rightarrow 4$ | $2 \rightarrow 5$ | 0.68956 | 0.68956 | 0.46721 |
| | | | $3 \rightarrow 7$ | $3 \rightarrow 8$ | | | |
| | | | $4 \rightarrow 12$ | $4 \rightarrow 11$ | | | |
| | | | $5 \rightarrow 13$ | $5 \rightarrow 12$ | | | |
| | | | | $6 \rightarrow 14$ | | | |
| | | | | $7 \rightarrow 15$ | | | |
| | | | | $8 \rightarrow 16$ | | | |
| | | | | $9 \rightarrow 17$ | | | |
| | | | | $10 \rightarrow 18$ | | | |
| | | | | $12 \rightarrow 19$ | | | |
| | | | | $13 \rightarrow 20$ | | | |
| ARQ | 500 | 0.001 | $2 \rightarrow 4$ | $2 \rightarrow 6$ | 0.54493 | 0.54490 | 0.27852 |
| | | | $3 \rightarrow 8$ | $3 \rightarrow 10$ | | | |
| | | | $4 \rightarrow 11$ | $4 \rightarrow 13$ | | | |
| | | | $5 \rightarrow 13$ | $5 \rightarrow 15$ | | | |
| | | | $6 \rightarrow 14$ | $6 \rightarrow 16$ | | | |
| | | | $7 \rightarrow 16$ | $7 \rightarrow 18$ | | | |
| | | | $9 \rightarrow 17$ | $8 \rightarrow 19$ | | | |
| | | | $13 \rightarrow 19$ | $9 \rightarrow 20$ | | | |
| | | | $19 \rightarrow 20$ | $10 \rightarrow 21$ | | | |
| | | | | $11 \rightarrow 22$ | | | |
| | | | | $13 \rightarrow 23$ | | | |
| ARQ | 500 | 0.005 | $3 \rightarrow 42$ | $7 \rightarrow 80$ | 0.052521 | 0.023069 | 0.022342 |
| | | | $4 \rightarrow 59$ | $8 \rightarrow 85$ | | | |
| | | | $5 \rightarrow 66$ | $9 \rightarrow 89$ | | | |
| | | | $6 \rightarrow 73$ | $10 \rightarrow 93$ | | | |
| | | | $7 \rightarrow 74$ | $11 \rightarrow 96$ | | | |
| | | | | $12 \rightarrow 99$ | | | |
| | | | | $13 \rightarrow 102$ | | | |
| | | | | $14 \rightarrow 105$ | | | |
| | | | | $15 \rightarrow 107$ | | | |
| | | | | $16 \rightarrow 109$ | | | |
| | | | | $17 \rightarrow 111$ | | | |
| ARQ | 700 | 0.0001 | $2 \rightarrow 6$ | $2 \rightarrow 6$ | 0.84380 | 0.84380 | 0.74753 |
| | | | $9 \rightarrow 7$ | $3 \rightarrow 10$ | | | |
| | | | | $4 \rightarrow 13$ | | | |

*continued on next page*

Table 3: Optimal value to protection mappings

| **Algo.** | **Size** | **BER** | **Opt. Map.** | **Local Map.** | **G. Opt.** | **L. Opt.** | **F. Opt** |
|---|---|---|---|---|---|---|---|
| ARQ | 700 | 0.0005 | 2 → 6 | 2 → 8 | 0.64914 | 0.64914 | 0.38704 |
| | | | 3 → 10 | 3 → 12 | | | |
| | | | 4 → 13 | 4 → 15 | | | |
| | | | 7 → 14 | 5 → 18 | | | |
| | | | 9 → 16 | 6 → 20 | | | |
| | | | | 7 → 22 | | | |
| | | | | 8 → 23 | | | |
| | | | | 9 → 24 | | | |
| | | | | 10 → 26 | | | |
| | | | | 11 → 27 | | | |
| | | | | 13 → 30 | | | |
| ARQ | 700 | 0.001 | 2 → 7 | 2 → 10 | 0.46202 | 0.46200 | 0.19933 |
| | | | 3 → 13 | 3 → 16 | | | |
| | | | 4 → 18 | 4 → 20 | | | |
| | | | 5 → 20 | 5 → 23 | | | |
| | | | 6 → 21 | 6 → 25 | | | |
| | | | 7 → 22 | 7 → 27 | | | |
| | | | 8 → 25 | 8 → 29 | | | |
| | | | 10 → 26 | 9 → 31 | | | |
| | | | 11 → 27 | 10 → 32 | | | |
| | | | | 11 → 33 | | | |
| | | | | 12 → 35 | | | |
| ARQ | 700 | 0.005 | 3 → 74 | 400 → 1 | 0.013425 | 0.0094942 | 0.0094942 |
| ARQ | 900 | 0.0001 | 2 → 7 | 2 → 8 | 0.84648 | 0.84648 | 0.72535 |
| | | | 3 → 8 | 3 → 13 | | | |
| ARQ | 900 | 0.0005 | 2 → 7 | 2 → 10 | 0.60039 | 0.60038 | 0.31995 |
| | | | 3 → 15 | 3 → 17 | | | |
| | | | 5 → 16 | 4 → 21 | | | |
| | | | 7 → 18 | 5 → 24 | | | |
| | | | 10 → 19 | 6 → 27 | | | |
| | | | | 7 → 29 | | | |
| | | | | 8 → 31 | | | |
| | | | | 9 → 33 | | | |
| | | | | 10 → 34 | | | |
| | | | | 11 → 36 | | | |

*continued on next page*

Table 3: Optimal value to protection mappings

| **Algo.** | **Size** | **BER** | **Opt. Map.** | **Local Map.** | **G. Opt.** | **L. Opt.** | **F. Opt** |
|---|---|---|---|---|---|---|---|
| ARQ | 900 | 0.001 | $2 \rightarrow 8$<br>$3 \rightarrow 18$<br>$4 \rightarrow 23$<br>$5 \rightarrow 30$<br>$8 \rightarrow 32$<br>$9 \rightarrow 34$<br>$12 \rightarrow 36$ | $2 \rightarrow 15$<br>$3 \rightarrow 22$<br>$4 \rightarrow 28$<br>$5 \rightarrow 32$<br>$6 \rightarrow 36$<br>$7 \rightarrow 39$<br>$8 \rightarrow 42$<br>$9 \rightarrow 44$<br>$10 \rightarrow 46$<br>$11 \rightarrow 48$<br>$12 \rightarrow 49$ | 0.38478 | 0.38475 | 0.14626 |
| ARQ | 900 | 0.005 | $5 \rightarrow 1$ | $400 \rightarrow 1$ | 0.0050318 | 0.0050318 | 0.0050318 |

## B.2 Random graph, tree build with Prim's algorithm

Table 4: Optimal value to protection mappings, random graph, minimum spanning tree build with Prim's algorithm

| **Algo.** | **Size** | **BER** | **Opt. Map.** | **Local Map.** | **G. Opt.** | **L. Opt.** | **F. Opt** |
|---|---|---|---|---|---|---|---|
| VG | 300 | 0.0001 | $3 \rightarrow 1$ | $3 \rightarrow 1$<br>$83 \rightarrow 2$ | 0.96316 | 0.96319 | 0.92818 |
| VG | 300 | 0.0005 | $1 \rightarrow 1$<br>$4 \rightarrow 2$<br>$47 \rightarrow 3$ | $1 \rightarrow 1$<br>$4 \rightarrow 2$<br>$44 \rightarrow 3$ | 0.91441 | 0.91441 | 0.90340 |
| VG | 300 | 0.001 | $1 \rightarrow 1$<br>$2 \rightarrow 2$<br>$7 \rightarrow 3$ | $1 \rightarrow 1$<br>$2 \rightarrow 2$<br>$7 \rightarrow 3$<br>$81 \rightarrow 4$ | 0.88755 | 0.88757 | 0.83384 |
| VG | 300 | 0.005 | $1 \rightarrow 4$<br>$2 \rightarrow 5$<br>$4 \rightarrow 6$<br>$12 \rightarrow 7$<br>$41 \rightarrow 8$ | $1 \rightarrow 4$<br>$2 \rightarrow 5$<br>$4 \rightarrow 6$<br>$11 \rightarrow 7$<br>$38 \rightarrow 8$<br>$145 \rightarrow 9$ | 0.80189 | 0.80188 | 0.76323 |
| VG | 500 | 0.0001 | $2 \rightarrow 1$<br>$22 \rightarrow 2$ | $2 \rightarrow 1$<br>$20 \rightarrow 2$ | 0.95226 | 0.95226 | 0.88188 |
| VG | 500 | 0.0005 | $1 \rightarrow 2$<br>$11 \rightarrow 3$ | $1 \rightarrow 2$<br>$11 \rightarrow 3$<br>$117 \rightarrow 4$ | 0.92516 | 0.92517 | 0.92363 |

*continued on next page*

Table 4: Optimal value to protection mappings, random graph, minimum spanning tree build with Prim’s algorithm

| Algo. | Size | BER | Opt. Map. | Local Map. | G. Opt. | L. Opt. | F. Opt |
|---|---|---|---|---|---|---|---|
| VG | 500 | 0.001 | $1 \to 2$<br>$2 \to 3$<br>$11 \to 4$<br>$83 \to 5$ | $1 \to 2$<br>$2 \to 3$<br>$10 \to 4$<br>$79 \to 5$ | 0.90937 | 0.90937 | 0.88882 |
| VG | 500 | 0.005 | $1 \to 6$<br>$2 \to 7$<br>$3 \to 8$<br>$8 \to 9$<br>$18 \to 10$<br>$56 \to 11$ | $1 \to 6$<br>$2 \to 7$<br>$3 \to 8$<br>$8 \to 9$<br>$17 \to 10$<br>$53 \to 11$<br>$172 \to 12$ | 0.82342 | 0.82342 | 0.78974 |
| VG | 700 | 0.0001 | $1 \to 1$<br>$8 \to 2$ | $1 \to 1$<br>$8 \to 2$<br>$297 \to 3$ | 0.96089 | 0.96089 | 0.95904 |
| VG | 700 | 0.0005 | $1 \to 2$<br>$4 \to 3$<br>$34 \to 4$ | $1 \to 2$<br>$4 \to 3$<br>$32 \to 4$<br>$328 \to 5$ | 0.93798 | 0.93798 | 0.93149 |
| VG | 700 | 0.001 | $1 \to 3$<br>$3 \to 4$<br>$16 \to 5$ | $1 \to 3$<br>$3 \to 4$<br>$16 \to 5$<br>$108 \to 6$ | 0.91923 | 0.91924 | 0.91184 |
| VG | 700 | 0.005 | $1 \to 8$<br>$2 \to 10$<br>$5 \to 11$<br>$16 \to 12$<br>$36 \to 13$<br>$102 \to 14$ | $1 \to 8$<br>$2 \to 10$<br>$5 \to 11$<br>$15 \to 12$<br>$34 \to 13$<br>$97 \to 14$<br>$294 \to 15$ | 0.83476 | 0.83476 | 0.80765 |
| VG | 900 | 0.0001 | $1 \to 1$<br>$5 \to 2$ | $1 \to 1$<br>$5 \to 2$<br>$116 \to 3$ | 0.96713 | 0.96714 | 0.96324 |
| VG | 900 | 0.0005 | $1 \to 2$<br>$2 \to 3$<br>$12 \to 4$ | $1 \to 2$<br>$2 \to 3$<br>$11 \to 4$<br>$105 \to 5$ | 0.94284 | 0.94284 | 0.92685 |

*continued on next page*

Table 4: Optimal value to protection mappings, random graph, minimum spanning tree build with Prim's algorithm

| **Algo.** | **Size** | **BER** | **Opt. Map.** | **Local Map.** | **G. Opt.** | **L. Opt.** | **F. Opt** |
|---|---|---|---|---|---|---|---|
| VG | 900 | 0.001 | 1 → 3<br>2 → 4<br>6 → 5<br>27 → 6 | 1 → 3<br>2 → 4<br>6 → 5<br>26 → 6<br>199 → 7 | 0.92491 | 0.92491 | 0.90356 |
| VG | 900 | 0.005 | 1 → 10<br>2 → 11<br>3 → 12<br>5 → 13<br>14 → 14<br>29 → 15<br>75 → 16 | 1 → 10<br>2 → 11<br>3 → 12<br>5 → 13<br>14 → 14<br>27 → 15<br>72 → 16<br>200 → 17 | 0.84420 | 0.84420 | 0.82406 |
| BCH | 300 | 0.0001 | 7 → 2 | 6 → 2 | 0.90522 | 0.90511 | 0.88059 |
| BCH | 300 | 0.0005 | 2 → 2 | 2 → 2 | 0.81060 | 0.81060 | 0.63942 |
| BCH | 300 | 0.001 | 1 → 2 | 1 → 2 | 0.79517 | 0.79517 | 0.79517 |
| BCH | 300 | 0.005 | 1 → 2<br>8 → 3<br>84 → 4 | 1 → 2<br>7 → 3<br>66 → 4 | 0.76156 | 0.76143 | 0.74834 |
| BCH | 500 | 0.0001 | 3 → 1 | 3 → 1 | 0.93163 | 0.93163 | 0.87399 |
| BCH | 500 | 0.0005 | 1 → 1<br>27 → 2 | 1 → 1<br>24 → 2 | 0.87579 | 0.87579 | 0.87431 |
| BCH | 500 | 0.001 | 1 → 1<br>7 → 2 | 1 → 1<br>7 → 2 | 0.85845 | 0.85845 | 0.84649 |
| BCH | 500 | 0.005 | 1 → 2<br>6 → 3<br>30 → 4 | 1 → 2<br>5 → 3<br>24 → 4<br>336 → 5 | 0.74414 | 0.74391 | 0.71846 |
| BCH | 700 | 0.0001 | 2 → 1 | 2 → 1<br>284 → 2 | 0.85947 | 0.85947 | 0.76494 |
| BCH | 700 | 0.0005 | 1 → 1<br>13 → 2 | 1 → 1<br>12 → 2 | 0.84645 | 0.84644 | 0.84324 |
| BCH | 700 | 0.001 | 1 → 1<br>4 → 2 | 1 → 1<br>4 → 2<br>240 → 3 | 0.82749 | 0.82749 | 0.80479 |

*continued on next page*

Table 4: Optimal value to protection mappings, random graph, minimum spanning tree build with Prim's algorithm

| Algo. | Size | BER | Opt. Map. | Local Map. | G. Opt. | L. Opt. | F. Opt |
|---|---|---|---|---|---|---|---|
| BCH | 700 | 0.005 | $1 \rightarrow 2$<br>$3 \rightarrow 3$<br>$31 \rightarrow 4$ | $1 \rightarrow 2$<br>$3 \rightarrow 3$<br>$26 \rightarrow 4$<br>$341 \rightarrow 5$ | 0.73530 | 0.73528 | 0.68984 |
| BCH | 900 | 0.0001 | $1 \rightarrow 1$ | $1 \rightarrow 1$<br>$358 \rightarrow 2$ | 0.89438 | 0.89438 | 0.89438 |
| BCH | 900 | 0.0005 | $1 \rightarrow 1$<br>$17 \rightarrow 2$ | $1 \rightarrow 1$<br>$15 \rightarrow 2$ | 0.88119 | 0.88116 | 0.87764 |
| BCH | 900 | 0.001 | $1 \rightarrow 1$<br>$5 \rightarrow 2$ | $1 \rightarrow 1$<br>$4 \rightarrow 2$<br>$149 \rightarrow 3$ | 0.85455 | 0.85441 | 0.82867 |
| BCH | 900 | 0.005 | $1 \rightarrow 2$<br>$2 \rightarrow 3$<br>$38 \rightarrow 4$ | $1 \rightarrow 2$<br>$2 \rightarrow 3$<br>$31 \rightarrow 4$<br>$102 \rightarrow 5$ | 0.72805 | 0.72802 | 0.66241 |
| ARQ | 300 | 0.0001 | $2 \rightarrow 2$<br>$3 \rightarrow 5$ | $2 \rightarrow 3$<br>$3 \rightarrow 4$<br>$4 \rightarrow 5$<br>$5 \rightarrow 6$<br>$6 \rightarrow 7$<br>$7 \rightarrow 8$<br>$9 \rightarrow 9$<br>$10 \rightarrow 10$ | 0.77147 | 0.77147 | 0.73737 |
| ARQ | 300 | 0.0005 | $2 \rightarrow 2$<br>$3 \rightarrow 4$<br>$4 \rightarrow 5$<br>$5 \rightarrow 6$<br>$6 \rightarrow 9$<br>$7 \rightarrow 10$ | $2 \rightarrow 3$<br>$3 \rightarrow 5$<br>$4 \rightarrow 6$<br>$5 \rightarrow 7$<br>$6 \rightarrow 8$<br>$7 \rightarrow 9$<br>$9 \rightarrow 10$<br>$11 \rightarrow 11$<br>$13 \rightarrow 12$<br>$17 \rightarrow 13$<br>$21 \rightarrow 14$ | 0.69476 | 0.69476 | 0.55647 |

*continued on next page*

Table 4: Optimal value to protection mappings, random graph, minimum spanning tree build with Prim's algorithm

| Algo. | Size | BER | Opt. Map. | Local Map. | G. Opt. | L. Opt. | F. Opt |
|---|---|---|---|---|---|---|---|
| ARQ | 300 | 0.001 | 2 → 2 | 2 → 3 | 0.60628 | 0.60625 | 0.39313 |
| | | | 3 → 4 | 3 → 5 | | | |
| | | | 4 → 6 | 4 → 7 | | | |
| | | | 5 → 7 | 5 → 8 | | | |
| | | | 6 → 8 | 6 → 9 | | | |
| | | | 7 → 9 | 7 → 10 | | | |
| | | | 8 → 10 | 9 → 11 | | | |
| | | | 11 → 13 | 11 → 12 | | | |
| | | | 16 → 14 | 13 → 13 | | | |
| | | | | 15 → 14 | | | |
| | | | | 19 → 15 | | | |
| ARQ | 300 | 0.005 | 2 → 5 | 3 → 15 | 0.17590 | 0.15550 | 0.032586 |
| | | | 3 → 11 | 4 → 19 | | | |
| | | | 4 → 15 | 5 → 22 | | | |
| | | | 5 → 18 | 6 → 24 | | | |
| | | | 6 → 20 | 7 → 25 | | | |
| | | | 7 → 22 | 8 → 27 | | | |
| | | | 8 → 24 | 9 → 28 | | | |
| | | | 9 → 25 | 10 → 29 | | | |
| | | | 10 → 26 | 11 → 30 | | | |
| | | | 11 → 27 | 12 → 31 | | | |
| | | | 12 → 29 | 13 → 32 | | | |
| ARQ | 500 | 0.0001 | 2 → 4 | 2 → 4 | 0.82616 | 0.82616 | 0.76539 |
| | | | 3 → 5 | 3 → 7 | | | |
| | | | 4 → 6 | 4 → 9 | | | |
| | | | | 5 → 12 | | | |
| ARQ | 500 | 0.0005 | 2 → 4 | 2 → 5 | 0.68687 | 0.68687 | 0.47378 |
| | | | 3 → 7 | 3 → 8 | | | |
| | | | 4 → 9 | 4 → 11 | | | |
| | | | 5 → 11 | 5 → 12 | | | |
| | | | 7 → 12 | 6 → 14 | | | |
| | | | 8 → 13 | 7 → 15 | | | |
| | | | | 8 → 16 | | | |
| | | | | 9 → 17 | | | |
| | | | | 10 → 18 | | | |
| | | | | 12 → 19 | | | |
| | | | | 13 → 20 | | | |

*continued on next page*

Table 4: Optimal value to protection mappings, random graph, minimum spanning tree build with Prim's algorithm

| Algo. | Size | BER | Opt. Map. | Local Map. | G. Opt. | L. Opt. | F. Opt |
|---|---|---|---|---|---|---|---|
| ARQ | 500 | 0.001 | $2 \rightarrow 4$ | $2 \rightarrow 6$ | 0.54016 | 0.54008 | 0.26442 |
| | | | $3 \rightarrow 8$ | $3 \rightarrow 10$ | | | |
| | | | $4 \rightarrow 11$ | $4 \rightarrow 13$ | | | |
| | | | $5 \rightarrow 13$ | $5 \rightarrow 15$ | | | |
| | | | $6 \rightarrow 16$ | $6 \rightarrow 16$ | | | |
| | | | $8 \rightarrow 20$ | $7 \rightarrow 18$ | | | |
| | | | | $8 \rightarrow 19$ | | | |
| | | | | $9 \rightarrow 20$ | | | |
| | | | | $10 \rightarrow 21$ | | | |
| | | | | $11 \rightarrow 22$ | | | |
| | | | | $13 \rightarrow 23$ | | | |
| ARQ | 500 | 0.005 | $2 \rightarrow 12$ | $7 \rightarrow 80$ | 0.065250 | 0.0079212 | 0.0079212 |
| | | | $3 \rightarrow 37$ | $8 \rightarrow 85$ | | | |
| | | | $4 \rightarrow 50$ | $9 \rightarrow 89$ | | | |
| | | | $5 \rightarrow 59$ | $10 \rightarrow 93$ | | | |
| | | | $6 \rightarrow 66$ | $11 \rightarrow 96$ | | | |
| | | | $7 \rightarrow 72$ | $12 \rightarrow 99$ | | | |
| | | | $8 \rightarrow 74$ | $13 \rightarrow 102$ | | | |
| | | | | $14 \rightarrow 105$ | | | |
| | | | | $15 \rightarrow 107$ | | | |
| | | | | $16 \rightarrow 109$ | | | |
| | | | | $17 \rightarrow 111$ | | | |
| ARQ | 700 | 0.0001 | $2 \rightarrow 5$ | $2 \rightarrow 6$ | 0.84320 | 0.84320 | 0.75730 |
| | | | $3 \rightarrow 6$ | $3 \rightarrow 10$ | | | |
| | | | $5 \rightarrow 7$ | $4 \rightarrow 13$ | | | |
| ARQ | 700 | 0.0005 | $2 \rightarrow 6$ | $2 \rightarrow 8$ | 0.64645 | 0.64645 | 0.38550 |
| | | | $3 \rightarrow 9$ | $3 \rightarrow 12$ | | | |
| | | | $4 \rightarrow 14$ | $4 \rightarrow 15$ | | | |
| | | | $5 \rightarrow 15$ | $5 \rightarrow 18$ | | | |
| | | | $9 \rightarrow 16$ | $6 \rightarrow 20$ | | | |
| | | | | $7 \rightarrow 22$ | | | |
| | | | | $8 \rightarrow 23$ | | | |
| | | | | $9 \rightarrow 24$ | | | |
| | | | | $10 \rightarrow 26$ | | | |
| | | | | $11 \rightarrow 27$ | | | |
| | | | | $13 \rightarrow 30$ | | | |

*continued on next page*

Table 4: Optimal value to protection mappings, random graph, minimum spanning tree build with Prim's algorithm

| Algo. | Size | BER | Opt. Map. | Local Map. | G. Opt. | L. Opt. | F. Opt |
|---|---|---|---|---|---|---|---|
| ARQ | 700 | 0.001 | 2 → 7 | 2 → 10 | 0.45756 | 0.45751 | 0.17213 |
| | | | 3 → 13 | 3 → 16 | | | |
| | | | 4 → 18 | 4 → 20 | | | |
| | | | 5 → 20 | 5 → 23 | | | |
| | | | 6 → 23 | 6 → 25 | | | |
| | | | 7 → 25 | 7 → 27 | | | |
| | | | 9 → 27 | 8 → 29 | | | |
| | | | | 9 → 31 | | | |
| | | | | 10 → 32 | | | |
| | | | | 11 → 33 | | | |
| | | | | 12 → 35 | | | |
| ARQ | 700 | 0.005 | 2 → 15 | 400 → 1 | 0.022488 | 0.0038576 | 0.0038576 |
| | | | 3 → 74 | | | | |
| ARQ | 900 | 0.0001 | 2 → 6 | 2 → 8 | 0.84587 | 0.84587 | 0.73658 |
| | | | 3 → 8 | 3 → 13 | | | |
| ARQ | 900 | 0.0005 | 2 → 7 | 2 → 10 | 0.59775 | 0.59775 | 0.30944 |
| | | | 3 → 13 | 3 → 17 | | | |
| | | | 4 → 15 | 4 → 21 | | | |
| | | | 5 → 16 | 5 → 24 | | | |
| | | | 6 → 17 | 6 → 27 | | | |
| | | | 9 → 18 | 7 → 29 | | | |
| | | | 10 → 19 | 8 → 31 | | | |
| | | | | 9 → 33 | | | |
| | | | | 10 → 34 | | | |
| | | | | 11 → 36 | | | |
| ARQ | 900 | 0.001 | 2 → 9 | 2 → 15 | 0.38079 | 0.38071 | 0.11222 |
| | | | 3 → 17 | 3 → 22 | | | |
| | | | 4 → 24 | 4 → 28 | | | |
| | | | 5 → 30 | 5 → 32 | | | |
| | | | 6 → 31 | 6 → 36 | | | |
| | | | 7 → 32 | 7 → 39 | | | |
| | | | 8 → 33 | 8 → 42 | | | |
| | | | 11 → 34 | 9 → 44 | | | |
| | | | 17 → 35 | 10 → 46 | | | |
| | | | | 11 → 48 | | | |
| | | | | 12 → 49 | | | |
| ARQ | 900 | 0.005 | 11 → 1 | 400 → 1 | 0.0029240 | 0.0029240 | 0.0029240 |

## C Size of Optimum in Parameter Space

The optimization problem is difficult, because the optimum is small in the parameter space.

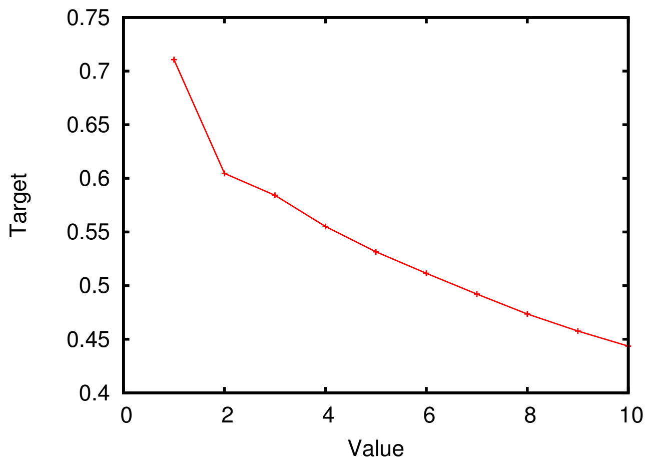

Figure 37: Value necessary to add a protection against a single bit error, BCH 63 BER 0.005, 300 bits “payload”

An example is shown in Figure 37. If the packet is protected against a single bit error as soon as it contains a single value, it reaches its maximum. If this is added later for higher values there is at first a steep decline. Later the curve is practically flat. This means that a slight deviation from the optimum leads to a sharp decline in performance. This emphasizes the need to operate in the optimum, but at the same time points out how difficult it is to find it.