Exploiting Wireless Link Dynamics

Von der Fakultät für Mathematik, Informatik und Naturwissenschaften der RWTH Aachen University zur Erlangung des akademischen Grades eines Doktors der Ingenieurwissenschaften genehmigte Dissertation

vorgelegt von

M.Sc. Software Systems Engineering

Muhammad Hamad Alizai

aus Haripur

Berichter:

Prof. Dr.-Ing. Klaus Wehrle
Prof. Dr.rer.nat. Bernhard Rumpe

Tag der Mündlichen Prüfung 07.03.2012

I hereby affirm that I composed this work independently and used no other than the specified sources and tools and that I marked all quotes as such.

Ich erkläre hiermit, dass ich die vorliegende Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.

Aachen, den 10. Januar 2012

Abstract

Efficient and reliable communication is the key to enable multihop wireless networks such as sensornets, meshnets and MANETs. Unlike wired networks, communication links in wireless networks are highly dynamic and pose additional challenges. Network protocols, besides establishing routing paths between two nodes, must overcome link dynamics and the resulting shift in the network topology. Hence, we need to develop efficient link estimation mechanisms, reliable routing algorithms, and stable addressing schemes to overcome these challenges inherent in wireless networks.

The prevalent approach today is to use routing techniques similar to those in wired networks, such as tree construction: Link estimators identify neighbors with consistently high quality links based on a certain cost metric. Routing protocols conserve routing to a single path between two communication nodes by choosing the best sequence of nodes at each hop, as identified by the link estimator. In contrast, recent studies on opportunistic routing schemes suggest that traditional routing may not be the best approach in wireless networks because it leaves out a potentially large set of neighbors with intermediate links offering significant routing progress. Fine-grained analysis of link qualities reveals that such intermediate links are bursty, i.e., alternate between reliable and unreliable periods of transmission.

We propose unconventional yet efficient approaches of link estimation, routing and addressing in multihop wireless networks to exploit wireless link dynamics instead of bypassing them for the sake of stability and reliability. The goal is to maximize routing performance parameters, such as transmission counts and throughput, by exploiting the burstiness of wireless links while, at the same time, preserving the stability and reliability of the existing mechanisms.

The contributions of this dissertation are manifold: Firstly, we develop relevant link estimation metrics to estimate link burstiness and identify intermediate links that can enhance the routing progress of a packet at each hop. Secondly, we propose adaptive routing extensions that enables the inclusion of such long-range intermediate links into the routing process. Thirdly, we devise a resilient addressing scheme to assign stable locations to nodes in challenging network conditions. Finally, we develop an evaluation platform that allows us to evaluate our prototypes across different classes of wireless networks, such as sensornets and meshnets, using a single implementation.

The dissertation primarily focuses on the *network layer* of the protocol stack. Although the proposed approaches have a broader relevance in the wireless domain, the design choices for our prototypes are tailored to sensornets – a notoriously difficult class of multihop wireless networks. Our evaluation highlights the key achievements of this work when compared to the state-of-the-art: The proposed metrics identify bursty links in the network with high accuracy, the routing extensions reduce the transmission count in the network by up to 40%, and the addressing scheme achieves 3-7 times more stable addressing even under challenging network conditions.

Zusammenfassung

Effiziente und verlässliche Kommunikation ist der Schlüssel, um Multihopkommunikation wie sensornets, meshnets und MANETs zu ermöglichen. Im Gegensatz zu kabelgebundenen Netzwerken sind die Kommunikationsverbindungen in kabellosen Netzen hochdynamisch und stellen weitere Herausforderungen dar. Netzwerkprotokolle müssen, neben der Vermittlung von Ende-zu-Ende Pfaden, auf diese Verbin-dungsvariabilität und die sich daraus ergebenden Änderungen der Netzwerktopologie reagieren. Folglich besteht ein Bedarf an effizienten Mechanismen zur Schätzung von Verbindungsparametern, verlässlichen Routingmechanismen und stabilen Adressier-ungsschemata, um diese inhärenten Herausforderungen kabelloser Netzwerke anzugehen.

Heutige Ansätze verwenden ähnliche Techniken wie kabelgebundene Netzwerke, zum Beispiel die Konstruktion von Bäumen: Verbindungsschätzer identifizieren Nachbarn, die nach einer vorgegebenen Metrik eine konsistent hohe Qualität zeigen. Routingprotokolle beschränken sich bei der Wahl einer Route auf einen einzelnen Pfad zwischen zwei Kommunikationsknoten, in dem sie bei jedem Schritt den nach dem Verbindungsschätzer besten Knoten wählen. Im Gegensatz dazu legen aktuelle Studien opportunistischer Routingschemata nahe, dass traditionelle Routingmechanismen suboptimal arbeiten, da sie Verbindungen mittlerer Qualität, die einen deutlich größeren Pfadfortschritt ermöglichen würden, aussparen. Eine genauere Untersuchung der Verbindungsparameter zeigt, dass diese Verbindungen mittlerer Qualität schubhaft sind, das heißt, dass sich Perioden verlässlicher mit Perioden unzuverlässiger Übertragung abwechseln.

In dieser Arbeit werden unkonventionelle aber effiziente Ansätze zur Verbindungsschätzung, zum Routing und zur Adressierung in Multihop-Netzwerken vorgeschlagen, die diese Dynamik in kabellosen Netzwerken für sich nutzen, anstatt sie zum Wohle von Pfadstabilität und Verlässlichkeit einzelner Verbindungen zu ignorieren. Das Ziel ist, Routingperformanzmetriken wie die Zahl der Übertragungen und den Durchsatz zu maximieren, indem man die Schubhaftigkeit kabelloser Verbindungen ausnutzt, während man gleichzeitig die Stabilität und Verlässlichkeit existierender Ansätze erhält.

Dieser Arbeit schlägt folgende Erweiterungen und Verbesserungen vor: Erstens wird eine Verbindungsmetrik entwickelt, mit Hilfe derer sich die Schubhaftigkeit von Verbindungen schätzen lässt und diejenigen identifiziert werden können, welche bei jedem Schritt den Pfadfortschritt verbessern können. Zweitens wird eine adaptive Routingerweiterung vorgeschlagen, die eine Miteinbeziehung dieser weitreichenden aber nur schubhaft zur Verfügung stehenden Verbindungen in bestehene Routingmechanismen ermöglicht. Drittens wird ein robustes Adressierungsschema vorgestellt, um Knoten unter dynamischen Netzwerkbedingungen stabile Adressen zuweisen zu können. Zuletzt wurde eine Evaluierungsplattform entwickelt, die es erlaubt, Prototypen über verschiedene Klassen von kabellosen Netzwerken, wie sensornets und meshnets, hinweg mit einer einzigen Implementierung zu untersuchen.

Der Fokus dieser Arbeit liegt primär auf der Netzwerkschicht des Protokollstapels. Obgleich die vorgeschlagenen Ansätze eine breitere Relevanz in kabellosen Netzwerken haben, orientieren sich die Designentscheidungen an sensornets, welche für ihre stringenten Herausforderungen bekannt sind. Unsere Evaluation hebt ein Schlüssel-

ergebnis beim Vergleich dieser Arbeit mit dem aktuellen Stand der Technik heraus: Die vorgeschlagenen Metriken identifizieren schubhaft verfügbare Verbindungen mit hoher Genauigkeit, die Routingerweiterungen reduzieren die Zahl der Übertragungen um 40%, und das Adressierungsschema erreicht eine um 3 bis 7-fach stabilere Adressierung selbst unter schwierigen und dynamischen Netzwerkbedingungen.

Acknowledgments

Who else to thank first but Allah, the most merciful and benevolent

I owe a lot to Professor Klaus Wehrle for keeping trust in me and making me believe as if I am one of his *special* students. Feeling special brings a lot of pressure as well because I always thought one day he will definitely get to know that I am not as intelligent and smart as he thinks. I am glad (or at least hope that) this day never came throughout my stay at ComSys. He hired me as a raw student and transformed me into a self-confident scientist who is willing to take on the strongest of the research challenges in his field. Thank you Klaus for *everything* that you did for me during my tenure both as a master and as a PhD student at ComSys. I will never forget that you prepared my IPSN 2010 poster on PAD. I am honored to have Professor Bernhard Rumpe as my 2nd PhD advisor and thankful to him for his highly valuable feedback, support, and prompt responses to my urgent requests.

Many thanks to my dearest ComSys colleagues (sorry guys, as much as I would loved to, I cannot mention your names because ComSys team has recently grown exponentially) for having to deal with my minimal German language skills, for interesting discussions and research colloquiums, for giving feedback on my talks and research papers, and of course for anything that I don't remember as of now. My coauthors Olaf, Jó, Hanno, Raimondas, Georg, Tobias V., Stefan, and Klaus made my papers look much better. Jó is one of the most amazing person I have ever met. I must thank him and Ismet for all the fun that we had together and I am sure that I will never be able to find roommates like them.

I am also highly indebted to ComSys thesis students Alexander, Tobias V., Benjamin, Bernhard and Andrea for their magnificent input in this dissertation work. These acknowledgements will be incomplete without thanking the technical and secretarial staff at ComSys because they helped me big time with all the administrative tasks.

Last but not the least, the never ending support and love since childhood and prayers of my parents are the reasons I am able to write this dissertation. I am thankful to my wife, for her continuous support and for being patient in tough days especially closer to paper submission deadline. How can I forget to thank my two year old son Abdul Ahad for his special treatment of my laptop (the only undamaged belonging at my home) and reminding me time and again that life is not all about doing research and writing papers.

Contents

1

Introduction

Wireless links vary significantly in their quality [ABB+04, CWPE05, CWK+05a]. Hence, unlike wired networks, the number of packets received by each neighboring node depends upon the quality of the link between a node and its neighbor. Several factors contribute to these variations among link qualities across a wireless network. For example, physical distance between a node and its neighbor [ZG03, BLKW08, ALWB08], environmental conditions [Rap01,PH06], and the interference experienced by each link from nearby networks operating in the same frequency range [HP09, KHC09, QZW+07, Nic07]. These link variations are well understood at the physical layer and have led to revolutionary developments in radio access technologies such as cellular networks.

However, the network protocols that use wireless medium do not understand these variations and cannot handle their implications [Sri10]. This is because these protocols are built on top of a link[1] abstraction which ignores the spatial and temporal properties of links. Consequently, network protocols tend to overlook these variations by limiting communications to only high quality and stable links. For example, routing protocols establish a tree based routing infrastructure where each node only communicates with its parent – typically a neighboring node with the best link quality and minimum hop distance to the root. This results in (1) a stable and clear-cut routing topology, (2) usage of short range links and little routing progress on each hop, and (3) heavy utilization of the selected links. We believe that this is not the optimal way to achieve multihop routing in a wireless network as it potentially ignores very useful links.

Previous studies [ABB+04, SKAL08, ZZHZ10] have shown that links follow certain patterns in their quality variations, especially the links with intermediate quality. For example, links show correlation in packet reception over time, i.e., they are bursty. We believe that by exploiting the underlying patterns of link variations, we can enable better utilization of links from routing perspective. This dissertation

[1] For simplicity, we use the term *link* as an abbreviation for *wireless link* throughout this dissertation

thus tries to explore these patterns, express them in the form of a protocol metric, and exploit them by developing relevant protocol extensions.

1.1 Problem Statement

Achieving multihop communication in a wireless network deals with three different mechanisms: (1) link estimation, (2) routing, and (3) addressing.

Link estimation is concerned with identifying high quality links within a node's one-hop neighborhood. These links are typically identified based on the long-term success rate of a link collected over a time frame in the order of minutes (or even hours) [BLKW08]. Although, in good network conditions, this approach is useful in maintaining a stable topology, this long term binding restricts a network in how well it can adapt to link dynamics. Hence, state-of-the-art link estimators are maladaptive in their operation. For example, in a sparse network with low density of nodes, a node might have no high-quality neighbor in its communication range, requiring a mechanism to deal with unstable connectivity. Similarly, today's link estimators are pessimistic in their link selection: They prefer short-ranged high-quality links over intermittently connected links that might reach farther into the network. Such links could offer better routing progress and hence reduce the number of transmissions, lower energy usage in the network, and increase throughput.

Routing protocols use link estimators to establish routing paths in the network that span multiple hops. A straightforward mechanism is to establish a tree-like topology by selecting the best quality link at each hop that minimizes the remaining distance (in hops) to the destination. We refer to this approach as *traditional routing* throughout this dissertation. Similar to link estimation, stability prevails over adaptability in today's routing protocols [RSBA07a]. It means that maintaining a stable routing tree is the ultimate goal of the existing routing algorithms. Hence, they are conservative in their path selection and only achieve suboptimal routing progress at each hop [RSBA07a, BM05a]. Their design is intentionally unable to realize fluctuations in the link qualities over a routing path. This is why they employ link estimators in the first place to identify long term stable links in the network.

Finally, an addressing scheme is required to achieve point to point communication in a multihop wireless network. A common scheme is to assign virtual coordinates to nodes: Construct multiple trees rooted at landmarks – designated nodes – and determine a node's location based on the vector of hop counts from a set of landmarks. The main challenge in such tree based addressing and routing schemes is that the changes at one node induce changes in all child nodes further down the tree. Hence, in unstable conditions, such schemes suffer heavily from rapid topological changes due to varying link conditions in the network. To cope with this challenge, maintaining trees and virtual coordinates across the network which are particularly consistent is understandably the main objective of these schemes. Therefore, they willingly concede performance penalties to achieve this objective.

1.2 Observations

A key assumption that implicates the basic design philosophy of today's wireless network protocols is that packet reception and packet loss events over a link are independent from each other [Sri10]. This relatively simple assumption has had a major influence in setting the functional level details of the three different mechanisms discussed in the previous section. For example, link estimators express the quality of a link by taking a moving average over a link's packet reception rates. Hence, no consideration, whatsoever, is given to the correlated packet loss events. This can be detrimental for the network performance if such loss events are relatively longer and go unnoticed at the routing layer. Similarly, this assumption implies that addressing and routing protocols cannot predict the fate of future transmissions over a link based on its very recent transmission history. Hence, it compels these protocols (1) to employ the naive method, i.e., use the best quality link at each hop, and (2) to avoid quick adaptation to the underlying link conditions as it leads to typical routing pathologies such as loops and network partitioning.

Our empirical observations contradict this assumption of independent packet losses over a link and thereby undermines many of the design decisions of today's routing protocols. Table 1.1 presents our key observations that form the basis of the concepts presented in this dissertation.

1.3 Major Contributions

While negating the underlying assumption of today's routing infrastructure in wireless networks, the observations in Table 1.1 lead to an important conclusion: Intermediate quality links are useful for enhancing the performance of today's routing protocols. However, the utility of such links for routing lies in three key questions:

- Can we define this correlation in packet reception over a link in terms of a metric that can be calculated at runtime?
- Can such a metric be used by routing protocols to include links with correlated packet reception (i.e., bursty links) for enhancing performance parameters, such as throughput and number of transmissions, without compromising the stringent stability and reliability requirements of today's applications?
- Can we formulate an addressing scheme that allows for inclusion of such links into the routing infrastructure while assigning stable locations to nodes?

This dissertation provides an affirmative answer to these questions by developing relevant mechanisms for all the three levels of wireless routing infrastructure. Moreover, this dissertation also demonstrates the generality of the presented mechanisms across multiple classes of wireless networks, such as IEEE 802.11 (Wi-Fi) and IEEE 802.15.4 (ZigBee).

Empirical observation	Comparison with today's assumptions	Implication on dissertation concept
More than 60% of the unused links in the network offer better routing progress than the links used by routing protocols.	Today's routing protocol only achieve suboptimal performance in terms of path stretch, i.e., number of hops enroute to destination.	Using such links could shorten the path stretch and thereby increase throughput and reduce the number of transmissions.
More than 70% of these unused links are bursty – alternate between reliable and unreliable transmission periods.	Packet loss events over a link are not necessarily independent.	Such links can be used for packet forwarding during their reliable transmission periods.
The probability of next transmission being successful over a link increases with the number of previous successful transmissions.	Protocols can predict, with high probability, the fate of future transmissions over a link.	We can possibly identify reliable transmission periods on a bursty link.
Due to unstable connectivity, a node's distance from a landmark vary significantly over time.	Assigning a static, current vector of hop counts leads to unstable addressing.	We need to find a mechanism that locates and addresses a node using variability patterns instead of an absolute vector.

Table 1.1 Key observations and their implications on the concepts presented in this dissertation. These observations are based on the empirical data collected from widely used wireless testbeds such as MoteLab [WASW05], Indriya [DCA09], Mirage [CBA+05], TWIST [HKWW06] and SWAN [Sta].

1.3.1 Link Estimation

The basic concept of our link estimation mechanism is to express the quality of a link in terms of how bursty it is. For this purpose we introduce two link metrics: First, we present MAC_3 – *Moving Average Conditional packet delivery function* – as a lightweight metric that estimates the burstiness of links based on the recent delivery traces at runtime. MAC_3 helps us in separating links that show correlated packet reception (i.e., bursty links) from the links that do not. Second, we define EFT – *Expected Future Transmissions* – as a metric to estimate the duration for which a bursty link remains reliable for transmission. EFT helps us in determining if the reliable transmission periods over a link are large enough to be used for packet forwarding. We also show that EFT is strongly correlated to MAC_3. Both these metrics are mandatory to determine whether or not an intermediate link is beneficial to the overall routing performance. Finally, based on these two metrics, we introduce a Bursty Link Estimator (BLE), derive requisite parameters for its usage, and evaluate its efficacy in estimating intermediate links. Our results indicate that BLE identifies bursty links in the network with high accuracy, hence paving the way for such links to be included into the routing infrastructure.

1.3.2 Routing

To effectively utilize BLE, we present Bursty Routing Extensions (BRE) that dynamically selects bursty links during the course of transmission. BRE describes a mechanism to implicity changes a node's parents without disrupting the underlying routing topology. Our evaluation on widely used testbeds indicate that BRE achieves an average of 19% and a maximum of 42% reduction in the number of transmissions when compared to other state-of-the-art proposals. Moreover, we show that both BLE and BRE are not tied to any specific routing protocol and integrate seamlessly with existing routing protocols and link estimators.

1.3.3 Addressing

We present a new addressing scheme, named Probabilistic ADdressing (PAD), that assigns probabilistic addresses to nodes. In PAD, a node learns from its past locations and calculates the probability distribution over its recent hop distances from landmarks. This probability distribution is then used as an address of the node and it incorporates all possible paths leading to landmarks. Hence, a node's location is defined in terms of the probability that it exists in a certain location and remains independent from the packet loss at shorter time scales. All other nodes in the network predict the current location of a node in its distribution. Our evaluation shows that PAD requires 3-7 times fewer address changes and even a simple routing strategy over PAD reduces the number of transmissions in the network by 26%.

1.3.4 Wi-Fi Evaluation

Finally, we show that the utility of BLE, BRE and PAD is not limited to any specific class of wireless network. Although our detailed protocol evaluation targets sensornets – a notoriously difficult class of wireless mesh networks – we prove the generality of our mechanisms by evaluating them across multiple classes of wireless networks. However, our goal is to avoid tedious re-implementation required to run protocols in different classes of wireless networks due to the lack of an integrated development environment. This typically restrict the developers to explore the feasibility of their protocols in only one class of wireless network and *implicitly* assume their applicability in the other [AKL+10, AWK+11a].

To this end, we introduce TinyWifi, a platform for executing native sensornets protocols on Linux-driven wireless devices. TinyWifi builds on nesC [GLvB+03a] code base that abstracts from TinyOS [LMG+04] and enables the execution of nesC-based protocols in Linux. Using TinyWifi as an evaluation and runtime platform, we demonstrate the superior performance of BRE, and PAD in IEEE 802.11 based networks as well.

1.4 Limitations

This dissertation also highlights the limitations of the proposed mechanisms. For example, BRE assumes dense deployments where a node has many neighbors to

choose from. Higher density of nodes increases the probability of finding a neighboring node with bursty link that offers better routing progress. Similarly, packet transmission rates also play a crucial role in determining the performance of BRE. This is because sending packets at higher rates over bursty links maintains a strong correlation between their success or failure providence. While by sending packets further apart, this correlation does not hold necessarily [Sri10].

The addressing scheme, i.e., PAD, is highly beneficial in challenging networking conditions with frequent variations in link qualities. However, it only performs as good as the state-of-the-art protocols in stable conditions dominated by good links. This is because in stable conditions both the probability distribution and the static vector of a node's hop distances from landmarks are almost identical.

We also discuss the memory usage, computational overhead and transmission cost of BLE, BRE and PAD. Each of these mechanisms offer a number of design choices and trade-offs between their efficiency and overhead. For example, PAD results in larger node addresses but allows to trade-off transmission overhead against memory overhead in how address information is disseminated in the network. The first option is to include a node's address in broadcast beacons which increases the beacon size. The second option is to only transmit a node's current hop distance from landmarks instead of the aggregated distribution. In this case, the neighbors that receive the beacon need to store a history of theses coordinates and compute the PAD address themselves.

1.5 Target Environments

Sensornets and meshnets provide flexible and robust ways of establishing network structures without the need for an exhaustive infrastructure. Routing structures in these networks are self-established and -maintained and depend on the presence of wireless links between nodes in the network. A resource-efficient utilization of these structures greatly increases throughput and network lifetime and reduces transmission energy and failures. Our work thus targets sensornets and meshnets due to their equivalent routing mechanisms. Although our analysis comprises empirical data from both IEEE 802.15.4 and 802.11 based wireless networks, the design choices are tailored to sensornets. This is because our prototypes and their evaluation targets this embedded class of wireless networks.

1.6 Structure

The remainder of this dissertation is structured as follows.

Chapter 2 provides background information by revisiting the fundamentals of link estimation, routing, and addressing concepts in wireless networks. It presents the state-of-the-art case studies in these three areas and qualitatively compares them with our proposed mechanisms to establish a formal background for later discussions.

Chapter 3 highlights the need for utilizing intermediate links in wireless routing. It introduces new link estimation metrics and presents the design and evaluation of our proposed link estimator (i.e., BLE) based on these metrics.

Chapter 4 presents the corresponding routing extensions (i.e., BRE) to enable the inclusion of intermediate links into the routing infrastructure. It highlights the associated challenges, such as concerning the stability and reliability of wireless routing, and how this dissertation addresses them. It also empirically compares an implementation of the proposed routing extensions with a state-of-the-art routing protocol in sensornets.

Chapter 5 presents a probabilistic addressing mechanism (i.e., PAD) to utilize intermediate links in point-to-point communication scenarios without compromising the stability of addresses. It evaluates the stability of our addressing scheme by considering different sources of dynamics in wireless networks, such as link variations and node failures.

Chapter 6 discusses two important contributions of this dissertation: First, it presents the detailed architecture of our proposed evaluation platform (i.e., TinyWifi) that enables direct execution of sensornet protocols on Linux based wireless nodes, such as in meshnets and MANETs. Second, it briefly evaluates the utility of the presented approaches in IEEE 802.11 based meshnets.

Chapter 7 concludes this dissertation and points to the future directions for this work.

2

Multihop Wireless Routing: Qualitative Perspective

In this chapter we revisit the fundamental concepts of link estimation, routing, and addressing in self-maintained multihop wireless networks. We present some of the prominent case studies that represent the state-of-the-art in these three areas. Although the discussion in this chapter covers a broad spectrum of wireless networking research, the case studies will pay special attention to sensornets. This is because our experimental evaluation targets sensornets. However, sensornets and meshnets also share inherent similarities: Common characteristics such as the need for multi-hop routing in mesh topology are pitted against challenges such as wireless link dynamics and node churn.

The goal is not just to introduce these studies but also to revisit their design philosophy in the light of our observations. We first examine the details of each case study at a requisite level to include the pivotal concepts in this dissertation. However, the core of this chapter deals with comparing these studies with the concepts presented later on. Hence, in the light of our observations (cf. Table 1.1), we try to make a case for the protocol extensions presented in the later chapters. In this regard, we define some key properties for each of the three areas and rate the case studies based on these properties.

Our comparison is only limited to a qualitative level for two reasons. First, the detailed quantitative comparison is deferred to later chapters until we present the complete design of our protocol extensions. Second, our comparative discussion targets the design philosophy of these protocols and not just their performance. For example, we are interested in comparing properties such as the scalability and reliability of a protocol design and not the achieved throughput of a particular implementation. Please note that our rating for different protocol properties is comparative and simply enables better understanding of the design tradeoffs among different approaches. This rating shall not be considered as a formal classification of the approaches discussed here.

Overall, we believe that this discussion forms the proper conceptual bases and facilitates a smooth sailing into the technical content that appears in later chapters. This chapter lays a formal background for this dissertation but does not explore all the competing solutions in our target areas: A related work section is devoted in each of the following chapters for this purpose.

The remainder of this chapter is structured as follows. In Section 2.1, we discuss link estimation and some of its prominent approaches. Section 2.2 discusses routing techniques by putting a special focus on sensornets. Finally, in Section 2.3, we present novel addressing mechanisms for self-maintained wireless networks.

2.1 Link Estimation

Link estimation is the first step towards building scalable and reliable multihop wireless routing structures. In this section, we discuss the basic concepts and requirements of link estimation. We also define the key properties of a link estimator to compare state-of-the-art studies.

2.1.1 Introduction

Link estimation deals with identifying high quality links in a wireless network. Depending upon a particular wireless domain such as sensornets and meshnets, the term high quality can be used to define a link that optimizes throughput, packet loss, congestion, routing progress, energy depletion, or any other form of routing performance measure. However, the predominantly used link metric employed by the majority of today's link estimators [FGJL07, DCABM05] is throughput. It is measured in terms of Packet Reception Rates (PRR) or, its reciprocal, Expected Transmission Count (ETX): the number of retransmissions required by a packet to reach its destination.

The main challenge in link estimation is that wireless links exhibit strong fluctuations in their quality, especially, when their quality is measured in terms of PRR. For example, Figure 2.1 shows that for *intermediate links* ($0.1 < PRR < 0.9$), these fluctuations strongly deviate from their *mean* values. Using such links for data transmission can be detrimental for the performance of a network. Hence, the main task of link estimation is to identify *good links* (PRR > 0.9) in the network and to limit packet transmissions to only a selected set of these links.

A *link estimator* estimates the quality of a link from recent transmission traces. The idea is to use transmission traces of sufficient length that minimizes the estimation error, i.e., keeps it within $\pm 10\%$ of the actual value. For a link to be scored, it has to be in the *neighbor table* maintained by link estimators. This is because a link estimator only stores transmission traces for links in its table. In order to facilitate scalable network structures, the size of this table is kept constant regardless of the node density. Similarly, other constraints may also apply depending upon the class of wireless network. For example, severe energy constraints in sensornets strongly limit the computational requirements and the transmission overhead of a link estimator.

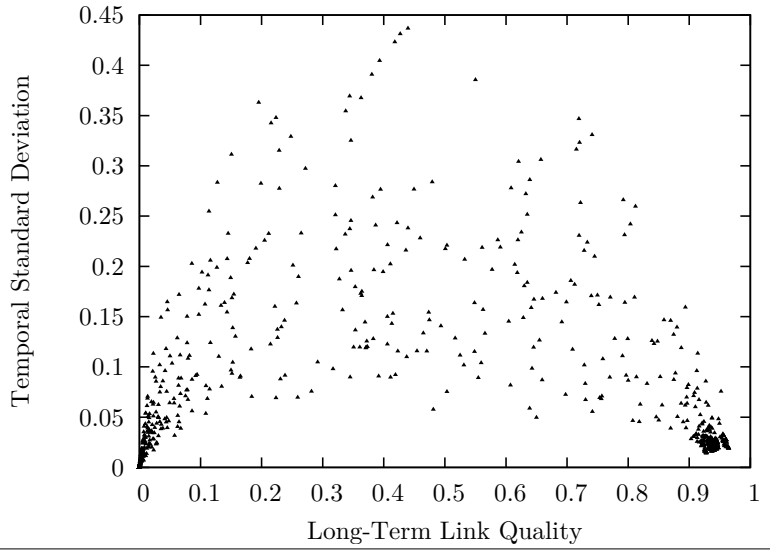


Figure 2.1 Wireless links exhibit inevitable fluctuations in their quality. The long-term link quality represent the PRR for entire experimental run. Each data point represents the standard deviation in PRR calculated over smaller time intervals for each directional node-pair. The graph shows data from an IEEE 802.15.4 based WSN deployment [ALWB08].

2.1.1.1 Table Management

Besides the accuracy of link quality estimates, an efficient strategy for neighbor table management is critical in expressing the performance of a link estimator. Table management typically deals with the following three operations.

Link Insertion: After receiving a packet from a neighboring node, the link estimator performs one of the following operations: (1) Update the link quality if the link already exists in the table, (2) insert the link in the table if the table is not already full, (3) ignore the link, or (4) evict a previous entry from the table and insert this new link. A link estimator has to carefully choose from these four options ensuring in the meantime that there are enough good links in the table that can be used for data transmissions.

Link Reinforcement: This operation deals with reinforcing the quality estimate of a link that already exists in the table. The thresholds for link reinforcement process, such as how often to perform it, has to be selected carefully to ensure that the newly calculated link quality does not overshoot or undershoot the desired accuracy threshold. Here the main tradeoff is between the agility and stability of the link quality estimates. Agility means assigning more weight to the recent estimates for adapting link quality to the most recent underlying link conditions. However, current link estimators prefer stability over agility by tuning parameters that control the history and the weight of the past estimates.

Link Eviction: Finally, a link estimator has to determine when to evict a link from the table. Commonly, a time-out or minimum data rate is associated with each link to detect node failures and evict corresponding links. Similarly, a minimum quality threshold is typically defined to evict links whose quality declines below that threshold. An efficient link eviction policy is important to evict unused links and make room for new, potentially valuable links in the table.

2.1.1.2 Key Properties

After discussing the basic operational details of link estimators, we now define key properties which, in our opinion, are essential for the design of a link estimator. These properties will help us rate the state-of-the-art techniques and shed light on their benefits and drawbacks.

- **Stability:** This property states the stability of the link estimator both in terms of link estimates and its ability to support a stable routing topology.
- **Adaptability:** It determines how quickly the link quality estimates converge to within the desired accuracy threshold and how well a link estimator adapts its tables to the underlying link dynamics.
- **Current Link State:** To see if the link quality reflects the exact link conditions at the time of data transmission or if it is only based on past statistics derived from periodic beacons. This is important because at the time of data transmission, networking conditions, such as traffic patterns and congestion, can be different.
- **Reception Correlation:** To determine if packet reception and loss events over a link are considered correlated or independent from each other.
- **Overhead:** The overhead introduced by a link estimator in terms of computational complexity, number of transmissions and packet overhearing. The transmission overhead includes active link beacons/signalling or additional link estimation information appended with each data packet. Moreover, packet overhearing also introduces significant overhead as a node has to receive and process packets that are not addressed to it.

2.1.2 Case Studies

We can divide current link estimation mechanisms into two broad categories, *long-term* and *short-term*.

In long-term link estimation, link qualities are estimated based on the delivery history of a link. We use the term *long-term* to emphasize that the focus of such link estimators lies on the long-term behavior of a link in the past. In a typical setting, each node snoops the channel for ongoing communication in a network, possibly both for periodic beacon packets and data transmissions. The packet loss over a link is inferred by assigning a unique sequence number to packets from each source. An ETX value is calculated over a window of size t: If n out of N packets are received during t then its ETX is N/n. Commonly, an *Exponentially Weighted Moving Average* (EWMA) is used over the past ETX values with α as a tuning parameter that controls the history of ETX averages. Nodes also exchange their link estimates with neighbors to aggregate bidirectional link quality. 4BLE [FGJL07], ETX [DCABM05] and BVR's link estimator [FRZ+05] are among the prominent derivatives of this method.

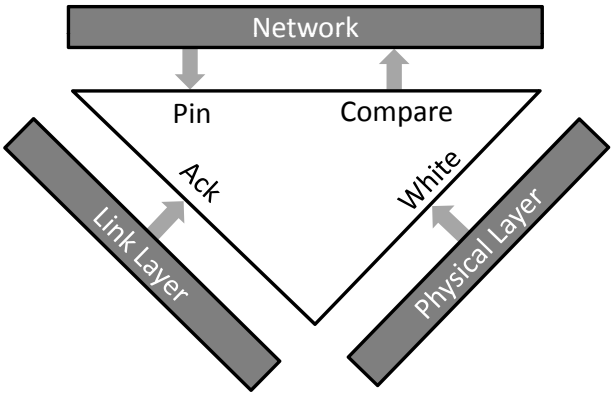


Figure 2.2 The 4BLE uses four bits of information: *Compare* and *Pin* bit from network layer, *Ack* bit from link layer, and *white* bit from physical layer to enhance unicast link estimates and table management policy [FGJL07].

On the other hand, short-term link estimation tries to predict the quality of a link based on instantaneous conditions. It does not necessarily maintain any recent history of a link but uses current link state (e.g., by sending active probes) to determine link availability at the time of data transmissions. The supporters of this mechanism argue that the link quality estimates derived from the transmission history of periodic beacon packets do not represent the current state of the link. For example, in sensornets the network traffic is generated by a rare occurrence of a nondeterministic event. Hence, the channel conditions, such as congestion, experienced by beacon packets in the past are completely different. SOFA [LKC06], STLE [BLKW08], LOF [ZAS09] and DUTCHY [PH08b] belong to this category of link estimation.

We now present a case study from each of these two categories.

2.1.2.1 Four-Bit Link Estimation

The Four Bit Link Estimator (4BLE) [FGJL07] is a state-of-the-art and classical example of a long-term link estimation. It couples link estimation information from broadcast beacons and unicast data transmission resulting into a hybrid ETX for each link. Moreover, 4BLE extends traditional ETX based estimation mechanism by combining information from three different layers – physical, link and network layers – to perform better table management. The key idea behind 4BLE is that each of these layers can provide useful information which benefits link estimation process. For example, the network layer can tell which links are most useful for routing and upper layer applications thereby facilitating a link estimator in link insertion and eviction decisions. Similarly, the physical layer can provide channel quality related information that helps a link estimator in distilling poor links from the estimation process. Overall, 4BLE defines four narrow interfaces to retrieve the following four bits of information – one from physical, one from link, and two from network layers (cf. Figure 2.2):

- **Pin:** The network layer can pin an entry in the table, preventing the link estimator from evicting this entry. This bit prevents the link estimator from evicting a useful entry from the table.

- **Compare:** It helps resolving inconsistencies between link estimation and routing tables. A link estimator can ask the network layer to compare a newly discovered link with an old entry in the table. The network layer responds by setting the compare bit to suggest that the route provided by the new link is better than the link already occupying the table. This bit helps a link estimator in identifying progressive link from routing perspective and estimate their quality instead of wasting critical resources over a useless link.

- **Ack:** This is the acknowledgement bit set in the transmit buffer if a packet transmission has been acknowledged by the receiver. This bit is used by the estimator to update the corresponding unicast link ETX.

- **White:** This bit reveals the channel quality per packet. A set white bit indicates high channel quality, which means that each bit in the packet has a very low decoding error probability. The white and compare bits are used conjointly to evict entries from the table: If the white bit for a packet received over a newly discovered link is set, the link estimator triggers the procedure corresponding to compare bit in order to decide if this link shall be inserted in the table by removing a random unpinned entry.

Rating: Figure 2.3 rates the performance of 4BLE against the properties discussed in Section 2.1.1.2. The rating scales from one (low) to five points (high). The positive or negative meaning of these scales depends upon the property itself. For example, in the case of scalability, a rating of one point means *poor* scalability, whereas, in the case of overhead, a rating of 1 point is interpreted as *very good*, indicating small overhead.

By relying on a long-term delivery history of a link of both broadcast beacons and unicast data transmission and extracting useful information from adjacent layers, the 4BLE is by far the most stable current link estimator. CTP [GFJ+09], a widely used collection protocol (cf. Section 2.2.2.1), uses 4BLE and outperforms contemporary approaches of routing in sensornets by maintaining a stable and a flawless topology. Therefore, it is assigned five points for stability.

4BLE uses an adaptive beaconing mechanism that increases beacon sending rate if a new node is added to the network or if routing inconsistencies (e.g., loops) are detected. This mechanism allows 4BLE to quickly converge its link estimates for a newly added link within the error threshold bounds. Similarly, 4BLE reacts quickly to link failures, i.e., after five failed data transmissions, by disqualifying the link for routing purposes. However, the adaptability of the 4BLE is only limited to such situations, it fails to quickly recognize the underlying link dynamics to improve performance [ALL+09, ALW12]: For example, if a previously ignored link becomes reliable and offers a significantly better alternative path than the current links in the

Stability	Adaptability	Current link state	Reception correlation	Overhead	Note
●●●●●	●●○○○	●○○○○	●○○○○	●●●○○	Long-term estimation

Figure 2.3 The performance rating and the use case for Four Bit Link Estimator.

table, 4BLE is unable to promptly react to such opportunities in the network. This is because there are no data transmissions occurring on this link and the adaptive beaconing slows down exponentially until there are inconsistencies detected in the network. Therefore, it is only assigned two points for adaptivity.

The link estimates in 4BLE are based on past delivery traces and does not regard the current state of the link. We still assign it one point because it actively monitors data path using *ack* bit and includes this information in calculating link estimates. In general, packet success and loss events over a link are considered independent from each other. However, it is assigned 1 point because it disqualifies a link after just five failed data transmissions. The overhead of 4BLE is moderate because (1) it only maintains a subset of neighboring nodes in the table for link estimation, and (2) uses active beaconing (link probes) to exchange link estimation information with other nodes. However, it does not employ packet overhearing and therefore is assigned 3 points.

2.1.2.2 Solicitation based Forwarding

Long-term link estimation tries to portray what can be expected from a link in the future based on how it behaved in the past. However, precisely this approach is its major drawback as well. For example, traffic patterns in sensornets are bursty: The network is in idle state most of the time and only generates large volumes of traffic when a certain event is detected in the environment. Hence, the link estimate derived from its past transmission statistics, i.e., when the network was in idle state, does not accurately reflect the actual quality of the link at the time of transmission. This is because the traffic patterns and congestion in the network are completely different at the time of traffic burst than when idle. Similarly, the active link probes transmitted when the network is in idle state are illusive and consume needless energy.

To address this problem, Lee *et. al.* present SOFA (SOlicitation based ForwArding) [LKC06] that uses a two way handshake to determine link availability. SOFA is not just a link estimator but a complete routing infrastructure for low-power wireless networks. However, its major contribution is the link estimation mechanism. The routing approach of SOFA is based on greedy hop-by-hop forwarding.

SOFA introduces a reactive two-way handshake protocol to determine link availability at the time of transmission. It does not maintain any other information (e.g., quality estimates) regarding a link. Each node, when needing to route a packet to sink, broadcasts a request called *Solicit-to-Forward* (STF). For example, in Figure 2.4, node A sends an STF received by its three neighbors B, C and D. A neighboring node receiving this message can respond with a reply message called *Accept-to-Forward* (ATF). In our example, C is the first node to reply with an ATF. After receiving this reply message, the sender node makes the replying node its *Designated-Next-Hop* (DNH) and starts forwarding its data as shown in the final step in Figure 2.4. The DNH is only determined on demand: A timer is associated with DNH and once a node is finished sending its packets for the current event and the timer expires, the node has to redetermine its DNH using the same handshake mechanism. SOFA also employs a passive acknowledgement mechanism: After forwarding a packet, the sender tries to overhear the transmission of its DNH. If it

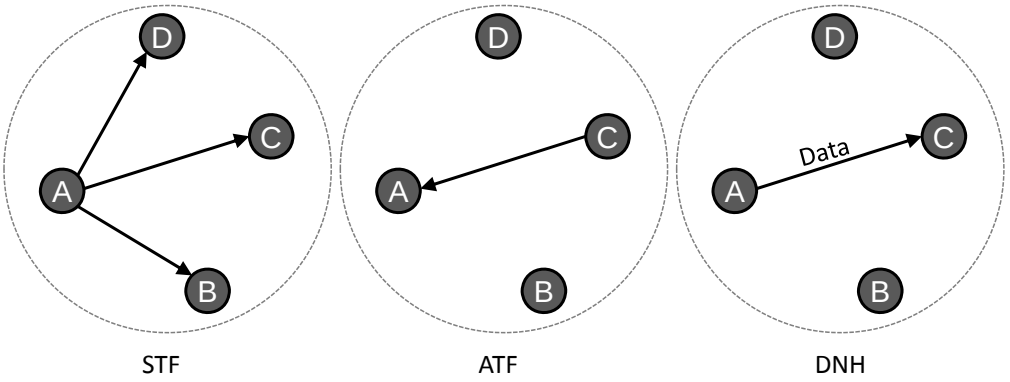


Figure 2.4 The two-way handshake in SOFA. Node A sends an STF message. Node C is the first neighbor to reply with ATF. Node A selects node C as its DNH and forwards data.

overhears the same packet that was recently forwarded, it implicitly assumes that the packet has been successfully delivered to the DNH. Otherwise, it retransmits the packet until DNH receives it or the maximum number of retransmissions are reached.

An important question is how does a node receiving an STF message determine if it is closer to the sink than the sender node. To this end SOFA assigns *height* to each node so that a receiver node can determine its location with respect to the sender node. The idea behind assigning heights is remarkably simple once understood. The sink node sends broadcast advertisements which are disseminated in the whole network. The advertisement is initialized with a height of zero (the sink node has zero height) and incremented by one at each hop as it propagates through the network. Hence, every node knows its relative distance from the sink node. A sender always sends its height in STF and a receiver only replies with ATF message if its height is less than the sender's. SOFA also employs *height maintenance* mechanism if inconsistencies are observed in the network. For example, if a node's height becomes a local minimum and no other node is replying with an ATF. This shall never happen in a fully connected network.

In dense networks there may exist a large number of neighbors that can reply with ATF. SOFA uses packet overhearing to limit the number of ATF responses. Consider an example with three nodes – A, B and C – each within the communication range of the other. Suppose node A wants to send data and thus broadcasts an STF, which is received by nodes B and C. Lets assume node B replies with an ATF before node C does. In this case, node C will snub its ATF because it has already overheard the ATF of node B. However, this mechanism only works if the neighboring nodes are within the communication range of each other.

Stability	Adaptability	Current link state	Reception correlation	Overhead	Note
●○○○○	●●○○○	●●●●●	●●○○○	●●●●○	Short-term estimation

Figure 2.5 The performance rating and use case for SOFA.

Rating: Figure 2.5 shows SOFA's rating. As oppose to 4BLE, SOFA does neither assigns link estimates nor maintains neighbor tables. Its only contribution towards stability is the height maintenance mechanism which remedies inconsistent topology due to high node churn in the network. Therefore, SOFA receives just one point for stability. The adaptability of SOFA is similar to 4BLE because it only adapts its link selection when *bad conditions*, such as lost transmissions or node failures, occur. However, it does not respond to the opportunities that appear during the course of transmission on other, potentially valuable links. For example, in SOFA, if a neighboring node offers better routing progress than the current DNH, the sender node will not change its DNH during a transmission burst. Therefore, it gets only two points for adaptability.

SOFA only uses the current link state information and hence it gets maximum points for this property. Similar to 4BLE, packet loss events are in general considered independent from each other by SOFA. However, the two-way handshake mechanism extrapolates a notion of packet reception correlation since the last packet delivery (i.e., STF packet) is considered sufficient for the success of succeeding transmissions (two points). Although SOFA does not require link tables for its operation, it has a very high communication overhead both in terms of the two-way handshake and the packet overhearing that consumes a significant amount of energy. Especially, the two-way handshake can be detrimental for network performance in challenging networking conditions when a node has to repeatedly select its DNH.

2.1.3 Qualitative Comparison with BLE

Both long-term and short-term link estimation mechanisms have their own advantages and disadvantages. 4BLE maintains a stable routing topology in the network at the cost of slow-adaptation to underlying link conditions, i.e., by ignoring progressive links that may become reliable during the course of transmission. On the other hand, SOFA uses the current link state for making its decisions, however, many of its design mechanisms are debatable. For example, in Section 3.4.2, we experimentally demonstrate that a single successful transmission cannot be considered as sufficient evidence of good link conditions. Moreover, its DNH selection mechanism is very inefficient: In the case of multiple nodes competing to become DNH, a sender node selects a neighboring node as DNH from which it receives the first ATF response. Hence, it ignores the possibility of using other potentially valuable neighbors.

BLE tries to combine the advantages and eliminate the disadvantages of both these techniques. We compare BLE with the existing mechanisms by rating it against our established criteria/properties in Figure 2.6. We argue that a stable routing topology is imperative for establishing reliable and robust routing structures. However, we show that a subtle design of a link estimator that explores transmission opportunities over long range intermediate links does not disrupt the stability of today's

Stability	Adaptability	Current link state	Reception correlation	Overhead	Note
●●●●●	●●●●○	●●●●●	●●●●○	●●●○○	Busty links

Figure 2.6 The performance rating and the use case for Bursty Link Estimator.

routing protocols. Therefore, BLE does not replace the long-term link estimation but serves as an additional and modular component that integrates well with existing long-term link estimation mechanisms. It allows the existing mechanisms to utilize communication opportunities that might arise over previously ignored class of links without disrupting the underlying routing topology (Stability = five points). The design and integration of BLE with existing link estimators is discussed in Section 3.5.

Similarly, BLE is optimistic in it link selection and allows a routing protocol to adapt to the underlying link conditions both in spreading *good news* and *bad news* to the neighboring nodes. The good news represents a situation where a long range intermediate link becomes temporarily reliable for transmission. BLE utilizes such opportunities as early as the next three transmissions. Similarly, the bad news represents a situation when an intermediate link again becomes unreliable for transmission. Based on empirical observations from multiple testbeds, BLE avoids overshooting an unreliable link by reverting back to a high quality link even if a single transmission fails over an intermediate link (Adaptability = four points, Current Link State = five points). BLE thresholds for qualifying or disqualifying a link are discussed in Section 3.4.2.

The key idea behind the development of BLE is to break the assumption of independent packet reception events over a link. It measures the quality of a link in terms of its burstiness that shows the correlation of packet reception events over a link (Reception Correlation= four points). This information is essential in determining if a link is useful for packet forwarding or not. To make this concept clear let us consider two links, one which rarely transmits a packet successfully and the other which alternates between reliable and unreliable transmission periods, i.e., it is bursty. Approaches such as SOFA cannot differentiate between these two links because they do not employ any mechanism to determine if the previous successful transmission occurred by chance or if this link is bursty. Similarly, it is unlikely that 4BLE will utilize this link because of its poor ETX estimate in the long-term. BLE's link estimation metrics are discussed in Section 3.4.

BLE is based on passive overhearing of packets and does not require active link probes. However, it is an extension rather than a replacement of existing long-term link estimation mechanism. Therefore, in addition to the underlying link estimator, such as 4BLE, it incurs additional overhead of packet overhearing and link estimate calculation (Overhead = four points). Section 3.5.3 gives details regarding BLE's overhead.

2.2 Routing

A link estimator is only concerned with a node's one hop neighborhood. Routing protocols establish multihop structures using link estimation information at each hop. In this section, we discuss some of the prominent routing approaches in sensornets. We also define key properties of a routing protocol and compare different approaches with our proposed extensions.

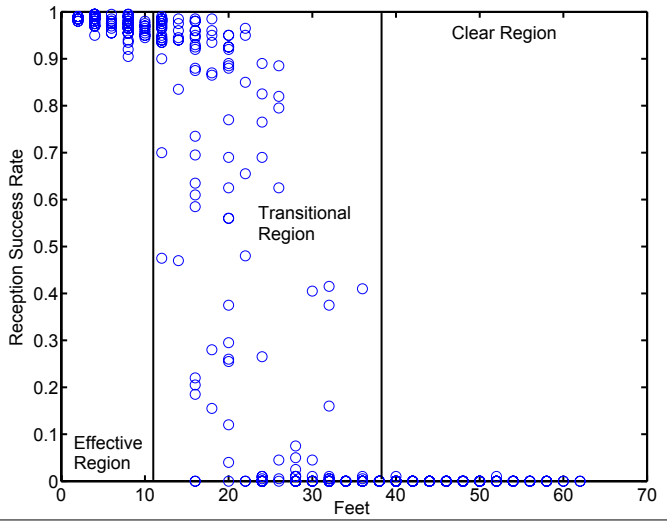


Figure 2.7 Reception rates vs. distance between nodes in a line topology: In the effective region all links exhibit good to perfect quality. The quality falls smoothly as the distance between nodes grow (transitional region) and eventually degrading to very poor link quality (clear region) [WTC03]

2.2.1 Introduction

Unlike wired networks, shortest path routing based on hop-distance metric is not feasible in wireless networks because a wireless link between two nodes reveals more dynamics than simply being considered available or not. For example, a link with $PRR = 40\%$ may deliver enough routing updates to be considered for data packets, thereby resulting in a significant number of retransmissions to deliver a packet. Figure 2.7 clarifies this observation further by showing the relationship between link reception quality and the distance between communicating nodes. Links from *transitional* and *clear* regions can dominate route selection because they offer better routing progress. However, using these links without assessing their reception quality leads to unstable routing topology, frequent retransmissions, and poor routing throughput.

In order to deal with these problems, contemporary routing protocols typically employ ETX [DCABM05] as a routing metric to establish high throughput paths between distant nodes. Path establishment in multihop wireless networks is usually based on distance vector routing approach: The participating nodes are not aware of the complete network topology. They only know the next hop that leads towards a particular destination and the routing cost along the path offered by that hop. Link state routing mechanisms have also been optimized for multihop wireless settings (e.g., OLSR [CJ03]), however, they are typically not preferred in large scale settings for two reasons, (1) limited scalability, and (2) inherent limitations of wireless devices (especially in sensornets) in terms of computations, storage and energy.

Routing approaches in wireless networks can be categorized in two broad categories, *address free* and *address based*. In address free routing, a node is typically assigned a unique identifier. It is mostly suited in situations where point to point communication is not relevant such as in data collection and dissemination in sensornets. Data flows in address free routing can be many-to-one or one-to-many. On the other hand, address based routing is needed for point-to-point communication scenarios

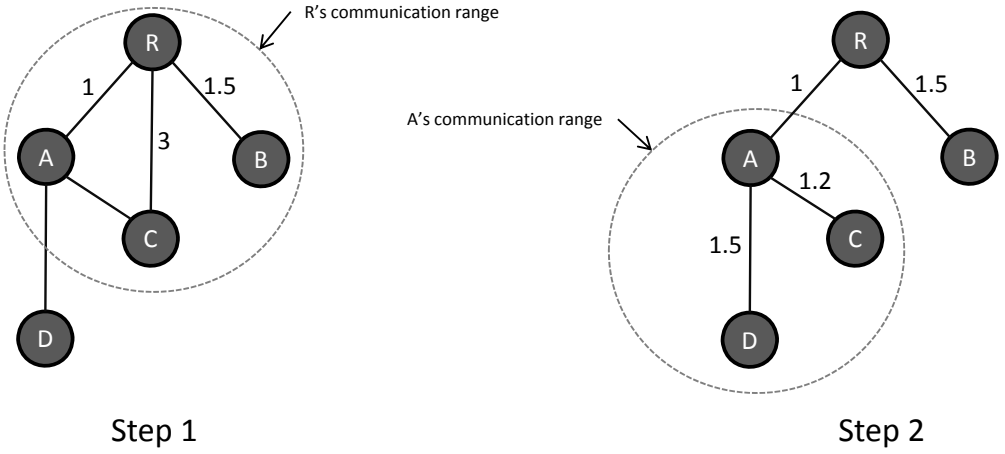


Figure 2.8 Tree construction example. The tree root R advertises itself with a distance of 0. Each node joins the tree by selecting a parent that minimizes the remaining cost (such as ETX) to the tree root.

where each node in the network can communicate with any other node. Nodes are usually assigned addresses that reveal their topological locations in a network. A vast majority of applications [ERS06,GEH03,LKGH03,DAG03] in sensornets require point-to-point communications.

We approach these two categories separately: In this section we focus on routing algorithms. The next section is devoted to addressing mechanisms that can be used with such routing algorithms.

2.2.1.1 Tree Construction

The majority of routing approaches in wireless networks are based on tree construction primitives. Especially, in networks with no access to location services, such as GPS, tree construction is at the helm of establishing scalable routing structures. However, tree construction based routing primitive is not a new concept: It is an established criteria even in wired networks, such as Internet back bones, which use the concept of sink trees and spanning trees for each participant in a multicast group [Tan02] .

Tree construction resembles the distance vector based routing mechanisms (e.g., Routing Information Protocol [Hed88]) where each node only maintains its one hop neighborhood and is unaware of the complete routing graph. For example, if a node X wants to send a packet to a distant node Z, it only knows that it can reach Z through its neighbor Y. However, it has no information, whatsoever, about the nodes on the remainder of the path from Y to Z.

We explain the tree construction phenomena by considering a simple example shown in Figure 2.8. A tree root R, i.e., a sink in sensornets or an Internet gateway in meshnets, advertises itself with a distance of 0. The distance can be represented by any metric of interest such as hop count or ETX. In this example, we consider ETX as a routing metric. Each node determines its bidirectional ETX from its neighbors using active link probes as discussed in the previous section. In the first step (cf. Figure 2.8), nodes A, B, and C receive this advertisement as they are within the

radio range of root R. As this direct link is the only choice currently available to reach R, in the next step, nodes A, B and C make R as their parent and replicate this advertisement, however, by respectively changing ETX values to 1, 1.5, and 3. In the second step, node D receives advertisements from both A and C and computes its path ETX as follows:

$$\textit{my ETX to neighbor <X> + ETX from <X> to R.} \tag{2.1}$$

Suppose both links $\overrightarrow{DA}$ and $\overrightarrow{DC}$ are of the same quality, D selects A as its immediate parent as the path over this node is clearly optimal. However, in this step C also receives the advertisement of A and realizes that using a single hop to reach R is more costly in terms of ETX than using A as a relay node. Therefore, it selects node A as its new parent and uses the new ETX value for subsequent advertisements. This process continues with the hope that ETX of a link will not change dramatically and a stable tree will be established with all nodes in the network joining the tree by selecting their parents. Tree construction suffers from typical distance-vector routing pathologies such as count to infinity, loops, and stranded nodes [Tan02]. Routing protocols employ mechanisms to recover from such pathologies. For example, loops are detected if a packet exceeds the maximum allowed number of hops specified in the *time-to-live* field.

2.2.1.2 Key Properties

Now we define key properties to establish a base for a fair qualitative comparison of routing case studies with our proposed routing extensions. Most of these properties are similar to link estimation properties discussed in the previous section, however, their definitions are extended at the network level instead of just a node's one hop neighborhood.

- **Stability:** Similar to link estimation, stability points to the steadiness of a routing topology and how gracefully a routing protocol recovers from node and link failures in the network.
- **Adaptability:** It determines how well a routing protocol adapts to the underlying link conditions, i.e., by responding to link estimator's suggestions, to enhance performance parameters such as throughput and number of transmissions.
- **Scalability:** It shows the maximum stretch of the routing topology in terms of how many nodes can be supported in the network without any communication breakdown. This is one of the most important properties for routing protocols in sensornets because the envisioned scale of deployment surpasses thousands of cooperating networked objects (motes).
- **Reliability:** The delivery rate of a routing protocol. It is one of the most important measure of routing performance in multihop wireless networks.
- **Overhead:** Routing overhead is measured in terms of transmission, i.e., the frequency and size of routing update messages, and storage, i.e., the memory required for maintaining routing structures such as routing tables.

2.2.2 Case Studies

From service point of view, we can divide routing protocols in wireless networks in two broad categories: *proactive* and *reactive*. As the name suggests, proactive routing protocols actively establish routing topology once a network is in place and a protocol is activated irrespective of if the applications really want to send data. Hence, such protocols maintain a connected network at any time. CTP [GFJ+09], MintRoute [HSNW10], OLSR [CJ03] are among the examples of proactive routing protocols.

On the other hand, reactive routing protocols are demand based and only establish a route when two nodes intend to communicate with each other. Once the communication is over, the routes are typically disabled after a certain period of time. These protocols are specifically useful in challenging networking conditions and mobility scenarios when maintaining an active routing topology is costly. DSR [JM96], AODV [PBRD03] and DYMO [BY09] are well known examples of reactive routing approaches.

We will now discuss three case studies: (1) CTP, state-of-the-art proactive routing in sensornets, (2) Opportunistic routing [BM05a, BM05b], a novel approach of exploiting link diversity in meshnets, and (3) AODV, a widely used reactive routing approach used both in sensornets and MANETs. Although this dissertation does not directly connect with reactive routing approaches, we present a case study here for completeness.

2.2.2.1 Collection Tree Protocol

CTP [GFJ+09] is one particular instance of collection tree protocol described in [RGJ+06]. It is state-of-the-art and one of the most widely used collection protocols shipped with TinyOS [LMG+04, LG09], an OS platform for sensornets implemented in nesC language [GLvB+03a, GLvB+03b]. It uses 4BLE as its link estimator.

The basic operational principle of CTP is the same as distance vector based tree construction discussed earlier. However, it uses some novel mechanisms to address two common problems of distance vector based approaches; (i) loops, and (ii) slow response to topological changes [Tan02]. The former is clear, however, the later requires some explanation. In distance vector routing, any news (e.g., node addition or breakdown), spreads across a network very slowly, one hop per update interval. So a node cannot be fully incorporated or removed from routing decisions until the news has spread across the whole network. Decreasing the update interval is a straight forward solution to spread the news quickly, however, it generates unnecessary traffic which is prohibitive in energy constrained sensornets. In order to address these problems, CTP introduces the following two mechanisms.

Datapath Validation: Typically, routing protocols use update messages to detect loops. However, CTP actively monitors data packets to solve any discrepancies along the data path. Each packet carries the transmitter's local ETX estimate to the destination, calculated using the mechanism discussed in Section 2.2.1.1. Logically, the ETX of the recipient node shall always be less than the ETX value in the received packet. This is because the transmitter will only send a packet to its parent that is

closer to the destination than itself. A packet is considered to be in loop if it violates this rule, i.e., its ETX is less than or equal to the receiver's ETX. Consequently, the receiver node initiates data path validation instead of simply dropping the packet. The data path validation deals with updating the ETX estimates of an out-of-date node using *adaptive beaconing*.

Adaptive Beaconing: As already mentioned, the sending frequency of routing updates (beacons) is a tradeoff between resource consumption and the recentness of the topology. CTP introduces an adaptive beaconing mechanism to strike an efficient tradeoff between the two. Using this mechanism, in emergency situations – such as addition/deletion of a node or loop detection – the network can respond within milliseconds by aggressive beaconing, while slowing it down significantly in normal conditions to save energy and bandwidth. The adaptive beaconing is a modification of Trickle [LPCS04] algorithm used for disseminating code updates in the network. In Trickle, a node suppresses its update and doubles the update-interval if it overhears a similar update, or decreases the update-interval to the minimum when it receives a new code update. Similarly, adaptive beaconing mechanism expands or shrinks a node's beaconing interval based on stable or unstable topological conditions in a network, respectively.

Rating: Figure 2.9 qualitatively evaluates CTP on functionality accounts. It is a very stable collection protocol based on long-term link estimation and efficiently repairs discrepancies in its routing topology. Therefore, we assign it four points for stability. The adaptivity of CTP stems from 4BLE: It changes a degrading link after just five failed transmissions. However, it is unable to use valuable opportunities on links that are black-listed by the link estimator due to their dynamic and bursty nature. Nonetheless, quick recovery of the topology using adaptive beaconing earns it two points.

CTP only maintains a constant number of neighbors, all one hop neighbors at maximum, in the routing table irrespective of the network size and node density. Therefore, it achieves high scalability (four points) as demonstrated by empirical evaluations in [GFJ+09]. The reliability of CTP is well proven as it has been thoroughly tested on twelve testbeds using six different link layer protocols [GFJ+09]. It delivered more than 90% of the packets on all testbeds with different physical topologies and varying link conditions, and hence, we assign it 4 points for its reliability. Finally, the overhead of CTP accounts for (1) the routing beacons exchanged among the neighboring nodes using the adaptive beaconing mechanism discussed earlier, and (2) the routing table, which maintains the state of a subset of one hop neighbors of a node.

Stability	Adaptability	Scalability	Reliability	Overhead	Note
●●●●○	●●○○○	●●●●○	●●●●○	●●●○○	Proactive collection

Figure 2.9 The performance rating and use case for CTP.

2.2.2.2 Opportunistic Routing

Opportunistic routing (or ExOR) [BM05a, BM05b] comes closest to BRE both in terms of how it operates, and its ambition of exploiting long range intermediate links in meshnets. Similar to BRE, ExOR does not operate as a stand alone routing protocol. It tries hard to forward packets over intermediate links that offer better routing progress and are closer to the destination. However, after delivering 90% of the packets in a *batch*, it uses the reliable delivery mechanism of an underlying routing protocol, such as OLSR, for delivering the remaining packets over the traditional path. In a similar fashion, BRE extends the ability of the underlying routing protocol to exploit intermediate links. There are two key differences between opportunistic routing and BRE: (1) The former uses broadcast primitive and the next forwarder (i.e., next hop) of the packet is determined among the receivers of a packet using an agreement protocol. The later uses unicast transmissions implying that the next forwarder of the packet is predetermined. (2) BRE aggressively reverts to traditional routing to avoid overshooting an unreliable intermediate link. However, ExOR operates on a batch of packets and tries very hard to deliver 90% of the packets before returning to traditional routing.

ExOR is based on the idea of cooperative diversity [vdM77] that uses broadcast transmissions to forward information through multiple relays. The destination can then use the best received signal or even combine information, i.e., to reconstruct the signal, received via multiple relays. ExOR utilizes two unique opportunities of link diversity in multihop wireless networks. First, using broadcast packet transmissions, it utilizes intermediate nodes along the traditional routing path to forward packets if the transmission falls short of the intended recipient. This way, the progress already made by a packet is utilized since an intermediate node, instead of the sender, forwards the packet further. Second, the packet may travel farther (e.g., 2 hop distance) than the intended recipient. ExOR makes use of this luck by providing mechanisms to allow farthermost recipient of the packet to become the next forwarder instead of the intended recipient.

Figure 2.10 explains the basic idea behind ExOR: Lets assume node A wants to send a packet to node D. In traditional routing, it forwards the packet to node C, the next hop in the routing table for node D. Suppose node C fails to receive this transmission but node B does. ExOR utilizes this opportunity by allowing node B to deliver this packet either directly to node D or via its next hop. Similarly, in the second case, the transmission from node A might occasionally be received by node D directly. ExOR also allows the routing protocols to take advantage of this good fortune.

In the following, we discuss the three main operational ingredients of the opportunistic routing protocol.

Determining the forwarder set: ExOR determines a prioritized subset of nodes that shall be responsible for receiving and forwarding the packet. To compute the forwarder set, ExOR requires knowledge about the loss rate of each link in the network. In the first step, a sender node calculates the shortest path to the destination. The first node in this path gets the higher priority to forward packets. The same procedure is repeated to complete the forwarder set by deleting the previously selected candidates from calculations and assigning lesser priority to the node that is selected

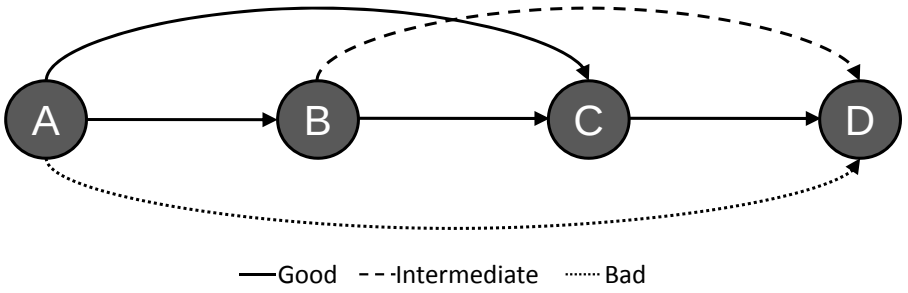


Figure 2.10 A simple example explaining the cooperative diversity utilized in opportunistic routing. Packets from node A to node C might occasionally be received by destination D directly or by node B. ExOR exploits such opportunities by avoiding retransmissions from node A.

in the later step. This forwarder set is then cached until the next link-state update. Each packet contains its forwarder list in the header.

Agreement protocol: The nodes in the forwarder set then use an agreement protocol to forward the packet. ExOR operates on batches of packets to minimize the overhead of the agreement protocol. The main purpose of the agreement protocol is to schedule the time when a node should transmit its fragment of the batch. Higher priority nodes, as indicated by the forwarder set, are allowed to transmit first. A node maintains a *forwarder timer* that is scheduled far ahead to allow higher priority nodes to transmit first. This timer is readjusted when the node overhears other node's transmission. Each node also maintains a *batch map* that determines, for each packet, the highest priority node known to have received that packet. The agreement protocol heavily relies on packet overhearing to update batch maps.

Reliable delivery: ExOR does not offer reliable delivery. Therefore, it uses the traditional routing as a backup mechanism, which employs hop-by-hop acknowledgements, for delivering the lost packets requested by the destination.

Rating: ExOR uses ETX based routing topology maintained by an underlying routing protocol. Therefore, we assign it 4 points for stability (as we did in the case of ETX based CTP's topology). Rating ExOR's adaptability is not straight forward: Although its performance is heavily dependent upon the underlying link condition, it does not pay any specific attention to varying link conditions at the routing layer. Nonetheless, it employs a highly efficient algorithm for packet forwarding that prioritizes the next hop selection; with the node closest to the destination always being assigned the highest priority. In short, ExOR's algorithm ensures that every progress made by a packet during a single transmission is utilized without taking link dynamics directly into consideration (Adaptability = 3 points).

ExOR doest not scale well because it needs link-state information of the whole network. Therefore, we assign it only 1 point for its limited scalability. ExOR itself

Stability	Adaptability	Scalability	Reliability	Overhead	Note
●●●●○	●●●○○	●○○○○	●●●●○	●●●●●	Cooperative diversity

Figure 2.11 The performance rating and use case for ExOR.

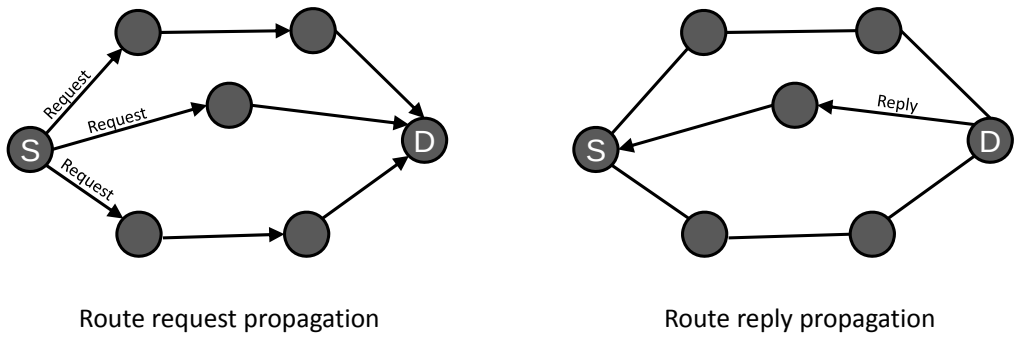


Figure 2.12 Route request and reply propagation through the network in AODV.

does not guarantee reliable delivery. However, the use of traditional routing as a backup ensures that it is at least as reliable as the traditional routing itself. Hence, it is assigned 4 points for reliability. The biggest limitation of ExOR is the overhead associated with its agreement protocol that includes (1) forwarder lists and batch maps appended with each transmitted packet, (2) packet overhearing to update node state, and (3) the computation complexity of the protocol itself. Therefore,we assigned 5 points for its high overhead.

2.2.2.3 AODV

In Section 2.2.2.1, we discussed CTP that has been specifically designed to meet the stringent resource constraints of sensornets. Similarly, in the previous section, we discussed ExOR that exploits wireless link diversity. However, ExOR is quite expensive both in terms of computational and transmission overhead and hence it is not suitable for resource constrained sensornets. AODV, on the other hand, is a more general reactive routing approach: It was originally designed for MANETs and later adapted to sensornets. Reactive routing approaches are useful in challenging network conditions where maintaining a consistent routing topology is expensive. For example, in a network with mobile nodes (e.g., MANETs) or limited connectivity between nodes due to harsh environmental conditions (e.g., sensornets). As this dissertation does not target reactive routing approaches, therefore, we will be very brief in our description of AODV. There are three steps in AODV's reactive routing approach, (1) route request (2) route reply, and (3) route maintenance.

Route Request: In AODV, each node maintains a small table containing information, such as a set of neighbors to forward packet to, for a particular destination. Links with neighbors are generally considered available or unavailable. Hence, it does not perform any active link estimation.

The route discovery in AODV proceeds as follows: When a source S wants to send a packet to destination D, it looks up its table to see if there is a neighbor entry for node D. Route discovery is only initiated if it does not find an entry in the table for node D. The same procedure is repeated at every intermediate node. When an entry for node D is not found, node S broadcasts a $ROUTE\_REQUEST$ packet relayed by all its neighbors until it reaches the destination or an intermediate node that already has an entry for node D in its table. Figure 2.12 explains this process by showing the paths taken by the $ROUTE\_REQUEST$ packet to reach node D.

Route Reply: When a route aware intermediate node or destination D receives the request it replies with a $ROUTE\_REPLY$ packet. However, this $ROUTE\_REPLY$ packet is now unicasted along the same path over which it was received – the smallest path is chosen if multiple requests are received – in the opposite direction (cf. Figure 2.12). As this packet traverses through the network, each intermediate node records an entry for node D for to establish a distance vector. Once this packet is received by the source of $ROUTE\_REQUEST$ packet, it initiates its data communication with node D.

Route Maintenance: Node mobility can cause sudden changes in the network topology. Therefore, a node has to keep track of which routes in the table are valid from time to time. In this regard, a node regularly exchange $HELLO$ messages, to which, each of its neighbor is suppose to respond. If a response message is not received, all the associate entries for the non responsive neighbors are deleted from the table.

Rating: AODV is a customized routing protocol only feasible in specific scenarios, i.e., networks with high node mobility. It does not match the stability of proactive routing approaches as routes are established based on simple route request and reply primitives. There is no link estimation performed and hence low quality links can dominate AODV's route selection without any particular consideration given to their feasibility for data transmission. Hence, it only receives 1 point for its stability. AODV treats links as available or unavailable, giving no consideration, whatsoever, to the underlying changes in link quality. It is assigned 1 point for its adaptability because of its route request and reply mechanism that allows for path reconstruction.

Longer tables with multiple forwarding candidates for each destination and an expensive route discovery mechanism strongly limit the scalability (1 point) of AODV. A number of studies [GSAP06, YIM+08, Kle08, DA10, CAAKA10] have been performed to implement and evaluate AODV in different classes of wireless networks under varying networking conditions. The reported reliability results of AODV differ significantly (e.g., from $< 50\%$ [Kle08] to $> 90\%$ [PRD99]) due to difference in evaluation environments. Similarly, the lack of link estimation makes it more susceptible to long range links of bad quality. The inclusion of such links results in frequent route discovery across the network due to frequent transmission failures on unreliable paths. Therefore, we assign it only 2 points for reliability.

Finally, it has a very high overhead (5 points) both in terms of memory footprint and bandwidth consumption due to frequent exchange of HELLO messages and route requests. However, it is clear that AODV targets specific ad hoc communication scenarios with high node mobility.

Stability	Adaptability	Scalability	Reliability	Overhead	Note
●○○○○	●○○○○	●○○○○	●●○○○	●●●●●	Reactive On-demand

Figure 2.13 The performance rating and use case for AODV.

2.2.3 Qualitative Comparison with BRE

Today's routing protocols in wireless networks use similar techniques as in wired networks [PB94, CJ03, BM05b, DPZ04]. They construct tree like topology and restrict communications to a very limited set of paths, typically a single path between two communicating nodes. These paths are constructed based on high quality links identified by a link estimator. Hence, today's routing approaches are pessimistic and conservative in their link selection and only achieve suboptimal routing progress [BM05b, RSBA07a]. This results in heavy utilization of a selected set of links and paths even though there is a multitude of other potentially useful paths available in the network.

BRE extends existing proactive routing approaches by providing relevant support to utilize link diversity inherent in wireless networks. It provides relevant support at the routing level to include intermediate links, i.e., recommended by BLE, in the routing process. In doing so, BRE allows traffic to be distributed among different links, relieving heavily congested paths and nodes. Similar to ExOR, BRE is opportunistic in its link selection and always prefer long range links before falling back to traditional routing algorithm. Hence, it would not be unrealistic to state that BRE is a simpler yet efficient variant of ExOR.

Figure 2.14 shows BRE's relative rating when compared to the existing routing approaches. BRE presents an adaptive routing strategy that allows a node to switch among different parents based on link estimation information while maintaining one primary parent as a backup. These changes in parents, however, are strictly local and do not impact the overall routing topology in the network. Therefore, it achieves the same stability as traditional proactive routing approaches (Stability = 4 points). Sudden changes in parent allows a node to adapt its next hop selection to very recent network conditions and exploit interesting opportunities over intermediate links. Nonetheless, BRE only promises optimal link selection within a node's one hop vicinity but does not promise optimal path in the network (Adaptability = 4 points). We discuss BRE's parent selection and adaptability in Sections 4.4.1 and 4.4.3, respectively.

BRE does not alter the stability of proactive routing approaches because it neither maintains any additional routing table nor exchanges further routing updates (Scalability = 4 points). Moreover, the design of BRE is highly modular and is not tied to any specific routing protocol. It integrates well with different routing approaches and link estimators. The scalable design of BRE is discussed in Section 4.4.

Similarly to ExOR, BRE maintains the packet delivery reliability of traditional routing approaches (Reliability = 4 points). In Section 4.5.3.2, we show that it even improves the packet delivery ratio of traditional routing in challenging network conditions by selecting best links based on instantaneous channel conditions. Finally,

Stability	Adaptability	Scalability	Reliability	Overhead	Note
●●●●○	●●●●○	●●●●○	●●●●○	●●●●○	Long-range forwarding

Figure 2.14 The performance rating and use case for Bursty Routing Extensions.

the only additional overhead introduced by BRE is its lightweight algorithm (Overhead = 4 points). We discuss BRE's overhead in Section 4.5.6.

2.3 Addressing

Point-to-point communications in multihop wireless networks require an addressing scheme to locate nodes in the network. Many addressing schemes have been proposed both for sensornets and meshnets such as geographical [IN99, BMSU99, KK00, KWZZ03], hierarchical [Tsu88a, Tsu88b, EFK07] and virtual coordinate addressing [CA06, FRZ+05, MOWW04, JS03, RRP+03]. However, our main focus lies in self-configurable and decentralized addressing schemes which are equally relevant in multiple classes of wireless networks. Therefore, in this section, we concentrate on addressing schemes that derive virtual node locations based on the underlying connectivity in a network.

2.3.1 Introduction

There are two main ingredients of point-to-point communication in multihop wireless networks, addressing and routing. Addressing deals with assigning locations to nodes in the network topology. A far located sender node uses this address for routing purposes. Routing on the other hand deals with actual decision making at each node to select the best next hop for the packet to reach its destination. In general, routing is performed greedily to allow for a scalable communication infrastructure that only requires a node to know its one hop neighborhood.

2.3.1.1 Challenges

Assigning locations to nodes in a multihop wireless network is a complicated task. As opposed to wired networks, there is no permanent network infrastructure that can be manually configured beforehand. Many factors contribute to rapidly changing topologies in a network such as node breakdown due to battery depletion in sensornets and a large number of participants moving, leaving, or joining the network in MANETs.

Traditional addressing schemes, such as IP, greatly suffer if applied to multihop wireless networks. For example, IP based hierarchical addressing is not feasible because it requires a very careful manual configuration of the entire network assuming a static topology. Unlike wired network, an IP address of a node in a wireless network is merely used to identify a node in the network for Internet communications but it does not reveal the routable location of the node. Another solution is geographical addressing that also requires either manual configuration or GPS support. However, in wireless networks, the connectivity graph is dynamic and strongly depends on the physical environment. Therefore, a geographical path leading towards a node might not be the optimal path based on the connectivity graph. Similarly, geographic routing suffers heavily from holes (or dead ends [YLRT09, LLM06]) in the network.

In recent years, virtual coordinates based addressing schemes have received much attraction in the research community for two main reasons: First, they are completely decentralized and self configuring. It means that nodes determine their addresses themselves after joining the network without any central coordination or manual configuration. Second, these schemes are based or the underlying connectivity graph, and hence, a node's address guides the packets to follow the best path leading towards the node. These benefits of virtual coordinate-based addressing mechanisms make them suitable for both sensornets and meshnets. Before presenting a few case studies on virtual coordinate-based addressing schemes, we identify the key properties of an addressing scheme in a wireless network.

2.3.1.2 Key Properties

Following are the key properties of an addressing scheme that we use to compare state-of-the-art case studies with our proposed approach PAD.

- **Address Stability:** This property states the number of times a node changes its address. Address changes may occur due to (1) variations in the underlying link conditions, or (2) frequent node failures. It is an important property because a node's location is typically stored in a distributed global database in the network and every change in the address requires an update in that database. Hence, address update is an expensive operation.
- **Address Monotony:** Once an address change occurs, this property determines the magnitude of difference (e.g., in hop counts) between a node's previous and new location. A smaller change in address (i.e., high address monotony) could result in higher routing success even if the packets are routed towards the destination using its outdated addresses. This is because the packets may still reach the vicinity of the destination whose new location is very close to the old one.
- **Resilience:** It shows how well an addressing scheme recovers from frequent node additions and departures from the network. In such dynamic scenarios, a resilient addressing scheme would require far less address updates in the network than a non-resilient one.
- **Scalability:** This is similar to routing scalability in the previous section. It shows the ability of the addressing scheme to enlarge itself to accommodate the growing number of nodes in the network.
- **Overhead:** It is measured in terms of storage requirements and control packets, such as beacons or address updates in the global database, exchanged to maintain stable addressing in the network.

2.3.2 Case Studies

In this section, we present two well known case studies of point-to-point routing, namely BVR [FRZ+05] and S4 [MWQ+10]. BVR is one instance of virtual coordinates based addressing specifically implemented for sensornets. S4 is a cluster based extension of BVR that achieves significantly smaller routing stretch than BVR.

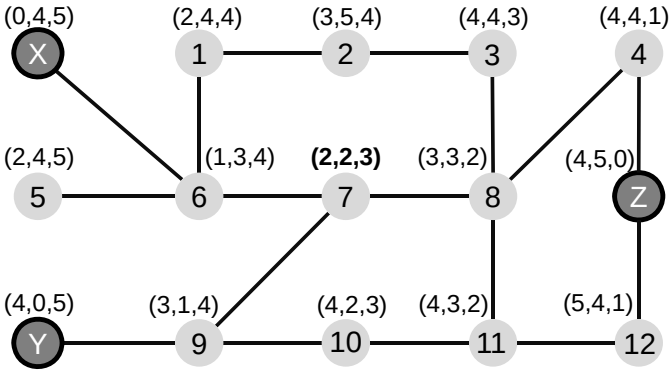


Figure 2.15 Virtual coordinates based addressing in BVR. Each node determines the hop distances from landmarks in the network. A vector of these hop distances, i.e., virtual coordinates, is used as a node's routable address.

2.3.2.1 Beacon Vector Routing (BVR)

BVR is also based on tree construction primitive. However, it needs multiple trees each rooted at *landmark*. A landmark is a designated node in the network used as a reference point by all other nodes. Every node in the network identifies its position in each landmark tree. The location of a node is defined in terms of a vector of hop distance from each landmark, commonly referred to as *virtual coordinates*. Routing is performed greedily over the virtual coordinates. There are two operational ingredients of BVR, virtual coordinate based addressing and routing.

Virtual Coordinates: Figure 2.15 shows an example of BVR's virtual coordinate based address establishment in a network with three tree roots (landmarks). Landmarks advertise themselves by repeatedly sending beacons. Based on these beacons, each node S (recursively) determines the number of hops $h(S, L_i)$ to each landmark L_i. The result can be viewed as a set of routing trees with the landmarks as their roots and with, for example, the hop count as a routing metric. A node S's coordinates $\vec{c}(S)$ in the virtual coordinate system are the λ-dimensional vector $< h(S, L_1), \ldots, h(S, L_\lambda) >$ with λ as the total number of landmarks. In our example in Figure 2.15, node 7 has a three-dimensional address vector $< 2, 2, 3 >$ where each vector component represents the node's hop distance to the landmarks X, Y, and Z, respectively.

Routing: Routing is performed greedily over these addresses. The idea is to let a node S choose a next hop T that minimizes the remaining distance $d(T, D)$ to the destination D (e.g., select a neighbor as a next hop whose coordinates are most similar to those of the destination node). BVR uses absolute component-wise difference as a routing metric:

$$d(T, D) = \sum_{i=1}^{\lambda} |T_i - D_i| \tag{2.2}$$

However, real-world deployments are confronted with lossy links that may falsely influence the hop distance from landmarks. It means that traversing one hop can require more than one transmission. Therefore, the "best" next hop is the one that results in the least number of transmissions necessary to reach the destination. BVR employs a link estimator to identify neighbors with stable links that minimizes ETX

for a successful delivery. Thus, only a selected subset of neighbors – offering an ETX below a certain threshold – are used in calculating the hop distance from the landmarks. Nonetheless, a node's address vector still represents the hop distance over the path with minimum ETX.

Rating: Figure 2.16 rates the performance of BVR. The address stability of BVR strongly depends upon the underlying network conditions. Tree construction offers a simple and attractive addressing solution, however, it is increasingly difficult to maintain stable trees in challenging network conditions. Changes in a particular node's coordinates propagate throughout the network and trigger further changes down the tree. For example, if a node close to a landmark changes its coordinate component for that landmark, all the descendant nodes will have to change their coordinates as well. Therefore, we only assign two points to BVR with regard to address stability.

The magnitude of change in node's coordinates (address monotony) is calculated by summing the absolute component-wise difference of each coordinate component. The idea is to see if changes in a node's coordinates are sudden or gradual. As BVR's tree construction process is based on long term link estimation, it strongly limits the number of options for reaching landmarks and this usually results in a higher magnitude of change in addresses. For example, a node may change its hop distance from two to four (magnitude of change = two points) for a certain landmark because this is the best option available among the set of limited neighbors with high quality links. Hence, BVR is only assigned 2 points for its address monotony.

BVR is not particularly resilient to address changes because it is unable to locally recover from node additions or failures [FRZ+05]. Thus, node dynamics lead to significant changes in the topology throughout the addressing tree (resilience = one point). The scalability of BVR is comparable to any other tree construction based routing approach, such as CTP. However, the state maintained per node is not constant and depends upon the number of landmarks in a network (scalability = three points). Besides state maintenance at each node, BVR relies on expensive packet overhearing based link estimator that appends link estimation information with each outgoing packet (Overhead = four points).

2.3.2.2 Small State and Small Routing (S4)

S4 is a cluster based extension of BVR that significantly reduces routing stretch. S4 argues that the attempts to maintain small state per node to achieve higher scalability can result in undesirable routing performance in terms of routing stretch – the ratio of the hop count of selected path to that of the optimal path [MWQ+10]. It minimizes both the state and routing stretch by combining the distance-vector based global network state and scoped distance-vector based local cluster state. S4

Stability	Monotony	Resilience	Scalability	Overhead	Note
●○○○○	●●○○○	●○○○○	●●●○○	●●●●○	Virtual coordinates

Figure 2.16 The performance rating and use case for Beacon Vector Routing.

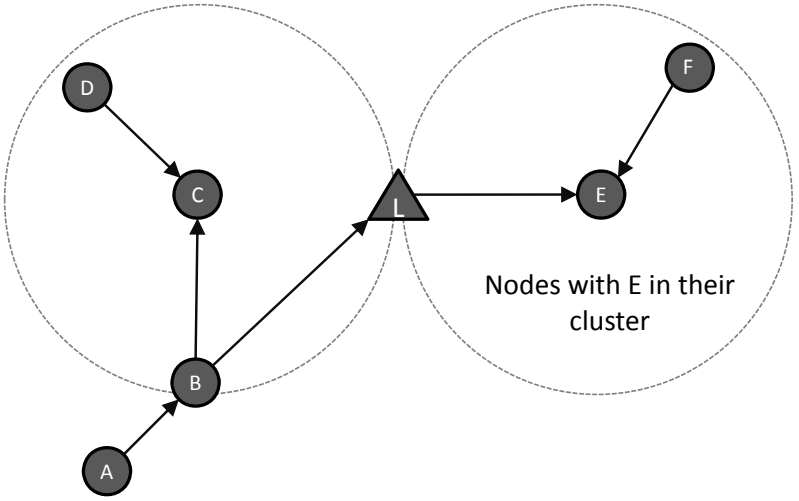


Figure 2.17 S4's routing scenarios. (1) A → C: B intercepts packets from A to deliver them directly to C instead of traversing through landmark L. (2) A → E: No shortcut is found and the packet is delivered via landmark (e.g., BVR's case). (3) D → C or F → E: Shortest path routing is used as the destinations are within the local cluster of sender nodes [MWQ+10].

achieves an average routing stretch of 1 and is the state-of-the-art point-to-point routing protocol in sensornets.

Algorithm: Apart from maintaining global virtual coordinates in the network, as in BVR, each node in S4 maintains a routing table for all the nodes in its local cluster. A node S's local cluster $C_k(S)$ contains all the nodes whose distance to D are within k times their distances to their closest landmarks. The idea behind maintaining a local cluster is that a node S can intercept packets addressed to node D and deliver them directly (cf. Figure 2.17). This significantly reduces the routing stretch because the packet does not have to reach the closest landmark of the destination, and then from there to the destination itself.

One of the key advantages of S4 is its small addresses. As opposed to BVR, the routing approach of S4 does not require the whole coordinate vector to be included as destination's address in the packet header. For packet forwarding, a node's address is nothing but the ID of its closest beacon. If a node wants to send a packet to another node within its cluster, it directly forwards the packet to the destination over the shortest path. However, if a sender node from another cluster sends a packet towards the destination's closest landmark, the packet is either intercepted by an intermediate node with destination in its cluster or it finally reaches the landmark and then delivered to the destination. Figure 2.17 depicts multiple routing situations and shows how S4 reacts in each of these situations.

Optimizations: Maintaining two level topological structure requires robust mechanism both for maintaining both inter-cluster and intra-cluster topology. S4 introduces relevant mechanisms to ensure a stable topology at both levels. For inter-cluster routing each node is supposed to know its (shortest-path) distance to all the landmarks in the network. Therefore, a reliable delivery of beacon packets (advertisements) initiated by landmarks is necessary to maintain a stable topology. This is because sudden packet losses can sometimes result in miscalculation of the distance while other times may require substantial changes in the topology, thereby

degrading the performance of S4. To address these challenges, S4 requires every node S in the network to rebroadcast beacons until n neighbors have received it or the maximum retransmission count t_{max} has reached. The choice of t_{max} and n is a tradeoff between the overhead and reliability.

Similarly, for intra-cluster routing, a node S has to retransmit a packet until an acknowledgment is received or the maximum retransmission count has been reached. In the later case, S initiates a local failure recovery request. After receiving this request, S's neighbors try to recover the packet locally. The idea is to select a node that is closest to the destination as the next hop for S. To avoid an explosion of local failure responses, in case a large number of S's neighbors maintain a distance vector for the destination in their tables, a prioritized response mechanism based on their distance vector is used. This way, each neighbor knows how long it has to wait before sending a failure recovery response.

Rating: The main difference between BVR and S4 is that the latter achieves smaller routing and transmission stretch. However, with regard to the properties defined for our qualitative comparison, S4 gathers a similar rating as BVR for scalability, monotony and resilience, as shown in Figure 2.18. Although S4 needs to maintain a global and local state per node, it can be as scalable as BVR by carefully selecting the state bounds. Similarly, address monotony remains the same because global virtual coordinates of S4 are based on BVR's distance vector approach. Recovering from node failures is at least as troublesome as in BVR. S4 achieves slightly higher stability than BVR because of its beacon rebroadcasting mechanism. Finally, the overhead of S4 is similar to BVR [MWQ+10].

2.3.3 Qualitative Comparison with PAD

Virtual coordinates, such as in BVR, offer an attractive addressing mechanism for multihop wireless networks whose deployments are often unplanned and lack any permanent network infrastructure. However, their direct adoption of tree construction primitive is not as efficient as in address-free collection protocols (cf. Section 2.2.2.1). This is because in virtual coordinate addressing both addressing and routing are strongly coupled with each other: A change in a node's immediate parent does not only impact the routing path towards a tree root but also the routable location of that node and all its descendants. Hence, link quality changes along a tree branch (path) force virtual coordinate based addressing mechanisms to recompute addresses of all the nodes connected to the tree via that branch. This limitation strongly impedes the routing performance despite high overhead for regular address updates in challenging network conditions.

In PAD, a node's virtual coordinates are expressed in the form of probability distributions. We introduce a degree of fuzziness in a node's address that acclimatizes

Stability	Monotony	Resilience	Scalability	Overhead	Note
●●○○○	●●○○○	●○○○○	●●●○○	●●●●○	Routing stretch

Figure 2.18 The performance rating and use case for S4 protocol.

short-term changes in the underlying link conditions. For example, if a node knows that it can reach a landmark in the network over multiple paths, it will not derive its coordinate component for that landmark by selecting the best path in terms of the offered quality and the number of hops. Rather, it will represent its coordinate component in the form of a probability function that expresses a subset of paths towards landmarks and the relative frequencies at which these paths are available. Hence, PAD doest not maintain any explicit trees in the network and automatically supports the inclusion of intermediate links into the routing process by embedding information regarding multiple paths leading towards a node in its address distribution. Thereby, a node's topological location is no longer dependent on a particular path but on a subset of such paths. As a result, link quality changes along a single path does not necessarily change the location of the nodes along that path since these nodes are reachable over multiple paths.

Figure 2.19 provides a comparative rating for PAD. One of the key advantages of PAD is its stability (4 points). Because it assigns fuzzy address to nodes instead of sharp coordinates, PAD concedes a degree of *error* in its addresses. Later in Section 5.4.3.1 we show that PAD achieves an order of magnitude higher address stability than BVR and S4. Section 5.4.2.2 discusses error tolerance in this fuzzy addressing scheme.

PAD achieves a very high address monotony (4 points) because it is neither dependent on stringent tree like topology nor on expensive link estimation. Hence, unlike BVR, which depends upon robust parent selection predominantly influenced by routing cost metric and link estimation, PAD lessens the need for this stringent parent-child relationship in a network. A node in PAD does not have a static position but a region where it can reside just like an electron resides in its region around the nucleus of an atom. As long as a node is within its assigned region, it can be reached without needing to change its coordinates. The same reasons apply for address resilience: Since a node is no longer dependent on a single parent, a sudden departure of a node does not necessarily impact the location of its descendant. However, in BVR, the importance of a node grows with regard to address resilience depending upon (1) how close it is to a landmark, and (2) how many descendants it holds in the tree. For example, the departure of a node closer to a landmark can break the whole routing tree associated with that landmark and inflict address changes throughout the network. We shed light on PAD's address establishment, resilience and monotony in Sections 5.3.2, 5.4.3 and 5.4.3.2, respectively.

Despite its long addresses, PAD achieves similar scalability as BVR and S4. First, because it only maintains a subset of routes leading towards landmark in its address distribution regardless of the node density (cf. Section 5.3.2). Second, because it offers a number of design choices with regard to address establishment, aggregation and dissemination in the network. For example, one such scalable design choice

Stability	Monotony	Resilience	Scalability	Overhead	Note
●●●●○	●●●●○	●●●●○	●●●○○	●●●●○	Fuzzy coordinates

Figure 2.19 The performance rating and use case for PAD.

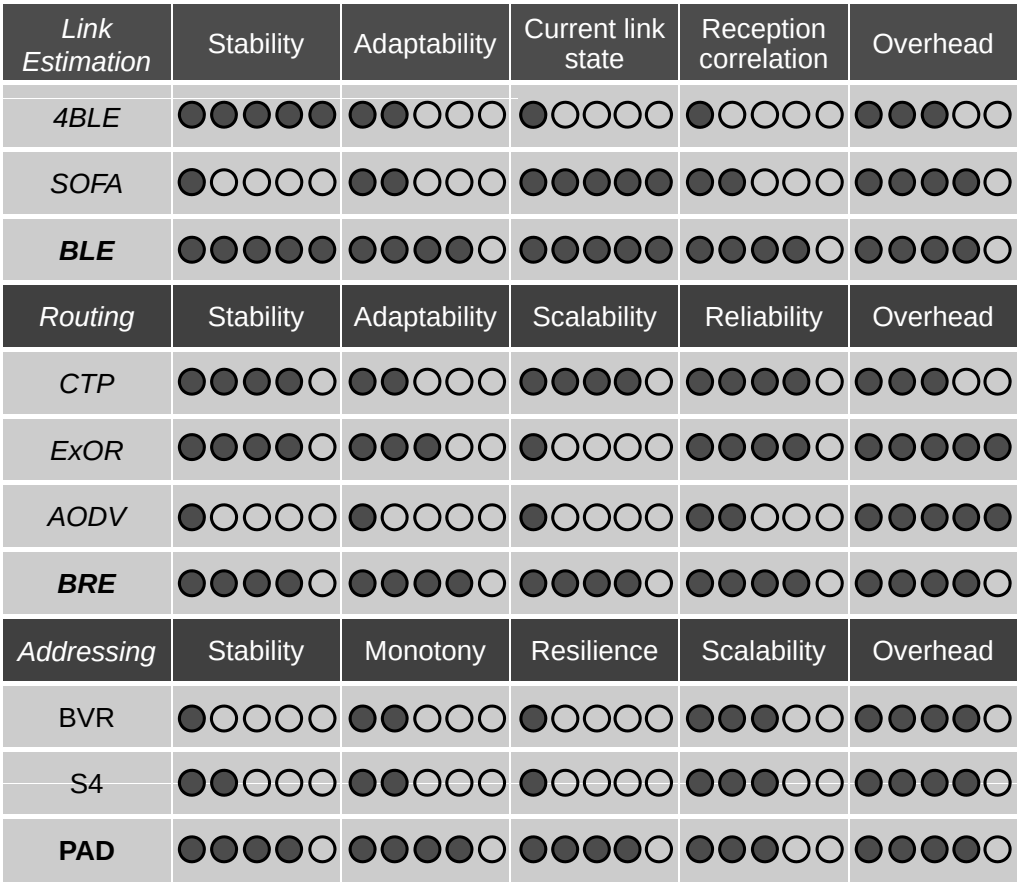


Figure 2.20 Summary of the performance rating assigned to case studies in the area of link estimation, routing, and addressing.

would be to aggregate PAD addresses in the form of *mean* or *weighted average* and use them for routing purposes. Section 5.5 discusses these concepts of aggregating PAD addresses.

The main overhead of PAD is its long addresses. However, when compared with BVR and S4, PAD neither employs packet overhearing nor link estimation for establishing addresses in the network. Therefore, we assign it a similar rating (Overhead = 4 points). The overhead of PAD is discussed in Section 5.6.4.

2.4 Summary

In this chapter, we discussed link estimation, routing, and addressing concepts in multihop wireless networks. We also presented state-of-the-art case studies from each of these three areas and conceptually compared them with the protocol extensions proposed in this dissertation. Figure 2.20 summarizes our comparison. We can clearly see that BLE, BRE, and PAD enhance different performance characteristics of existing approaches.

Long term link estimation is the preferred mechanism employed by today's link estimators. Its primary goal is to establish a stable routing topology. However, in achieving this goal, it mainly disregards packet reception correlation and the current state of a link at the time of packet transmission. 4BLE adds a degree of adaptiveness

to this estimation technique by demoting a link immediately after five consecutive packet failures. This helps in improving routing reliability but contributes little towards our goal of utilizing long range intermediate links in the network. Contrarily, short term link estimation primarily focuses on the current link state but achieves limited stability – a primary routing design requirement. BLE tries to combine the advantages of both these approaches. It does not underestimate the need of stable routing topology, while at the same, provides relevant mechanisms to estimate long range intermediate links and utilize them for packet forwarding.

Routing protocols typically utilize high quality links for packet forwarding. The idea is to convert the network graph into a simplistic tree like structure and only use the links that form the branches of that tree. In doing so, they limit packet forwarding to a very limited set of links. ExOR provides an elegant solution to efficiently utilize link diversity and the broadcast nature of wireless medium. However, its computational requirements and reliance on link-state information for each directed link in the network limits its usage to resource rich platforms such as in meshnets. BRE provides an alternative and light-weight solution to exploit link diversity. It has a transparent design that can integrate well with existing routing approaches and does not introduce any changes in the protocol building blocks such as packet headers.

Virtual coordinate based point-to-point routing approaches are also unable to exploit link diversity to ensure long term stable addressing in the network. Despite a tremendous emphasis on address stability, these approaches suffer from frequent address updates in dynamic network conditions. In this regard, PAD provides a sophisticated solution to address both these problems. A PAD address is composed of multiple paths leading towards a node and also exposes the quality of these paths in the form of a probability distribution. Moreover, it assign fuzzy locations to nodes to account for sudden changes in link conditions and thus maintain stable addressing across the network even under challenging network conditions. In the following chapters we discuss the design and evaluation of BLE, BRE and PAD in detail.

3

Estimating Link Burstiness

After introducing the problem space and establishing a formal discussion background in the previous chapters, we now turn our focus towards the actual contributions of this dissertation. We begin with our first contribution, i.e., link estimation, which forms the basis for including intermediate links into the routing process.

In the previous chapter, we learned that the prevalent approach in proactive routing algorithms is to employ a link estimator that identifies high quality links for packet transmissions. The use of intermediate links is thus disregarded (except if there are no high quality links in a network), although these links provide considerable additional resources for routing. This is because sudden changes in their transmission success rate make it difficult to accurately estimate the quality of such links and predict the fate of future transmissions [MRBT08]. Another reason for the exclusion of intermediate links from the routing process is their poor ETX estimates calculated over a longer period of time. We argue that ETX is not an optimal metric to express the nature of such links at short time scales. In particular, *bursty links* pose a major challenge to existing link estimation mechanisms [AWK+11b]. Estimating link burstiness and the length of successful transmission bursts is pivotal in assessing the utility of these links from a routing perspective. Therefore, we need a specialized link estimator to estimate intermediate links.

Based on significant empirical evidence of over 100,000 transmissions over each link in widely used IEEE 802.15.4 and IEEE 802.11 testbeds, we propose two metrics, EFT and MAC_3, for runtime estimation of bursty wireless links. We introduce a new link estimator (BLE) that, based on these two metrics, accurately estimates bursty links in a network rendering them available for packet transmissions. BLE is optimistic in its link selection and prefers long-range intermediate links over short-range stable links.

The rest of this chapter is structured as follows. Firstly, we motivate the problem space and discuss related work in Sections 3.1 and 3.2, respectively. We then analyze and define the exact scope of our work in Section 3.3. From this, we derive the design of our metrics and show their viability in Section 3.4. Finally, Section 3.5 presents

the design and evaluation of our link estimator before we summarize the discussion in Section 3.6.

3.1 Motivation

The availability and reliability of wireless links exhibit dynamic behavior at short and long time scales [LCL07, KCPnC09]. Therefore, choosing the best link, in terms of routing progress and the need for transmission resources, requires an accurate and timely estimation of the available links. Current link estimators, using metrics like PRR and ETX, only capture link dynamics at long time scales for the sake of a stable routing topology. These metrics estimate the quality of a link over extended periods of time – in the order of minutes – and thus achieve poor estimates for rapidly changing bursty links. As a result, such links are typically excluded from the routing process. However, recent protocol studies [ALL+09, BM05a, WTC03, PH08a] demonstrate that these links are long range and achieve significantly higher routing progress than stable links. Using these links therefore covers otherwise multiple transmissions and thus saves the energy and resource consumption coupled with these transmissions. Furthermore, previous studies [Zan97, SDTL06a] have shown that typical traffic patterns in the Internet as well as in multihop wireless networks are bursty. Hence, an optimal online link estimation at the time of a burst benefits spontaneous transmissions as well as the overall network performance.

Link burstiness is a well established fact: It has been thoroughly analyzed [ABB+04], accurately modeled [KCPnC09], and experimentally measured [SKAL08]. In bursty links, shifts between phases of reliable and poor packet delivery occur at short time scales, but future packet delivery is correlated to the recent success rate. Despite establishing a very strong knowledge base regarding the causes of link burstiness over the past few years, we still lack metrics that define the quality and usability of bursty links. Similarly, we need a link estimator that can assess link usability online (i.e., at runtime) to enable the inclusion of these links in the routing process.

3.1.1 Link Categorization

Until now we have been using indeterminate terms, such as *good* or *bad* links, to refer to the different types of links based on their transmission quality. Before digging into the details of our link metrics, we believe that it is essential to formally define these terms for the sake of better understanding of the concepts presented in this dissertation. All the link categories defined in Table 3.1.1 are not new. Similar terms have been used in the literature previously [SKAL08, Sri10].

The three categories that we specifically define for this dissertation are *bursty*, *independent* and *unused* links. To the best of our knowledge, the only formal definition of link burstiness is provided by Srinivasan *et. al.* [SKAL08]. Our definition of link burstiness slightly differs: We want to *estimate* link burstiness at runtime, while they try to *measure* link burstiness offline for understanding link behavior to fine tuning protocol parameters. Hence, our definition of a bursty link is biased towards (1) the utility of a link for packet transmission, and (2) a metric that can be calculated at protocol runtime. Therefore, we define a link as *bursty* if one can predict

Category	Definition
Good	PRR $>$ 90%
Intermediate	10% $<$ PRR $<$ 90%
Bad	PRR $<$ 10%
Bursty	CPDF(3) $>$ 75%
Independent	CPDF(3) $<$ 75%
Unused	Any of the above links not used for routing purposes

Table 3.1 Link categorization: A link estimator tries to identify good links in a network. Bursty links show correlated packet delivery and one can predict the fate of future transmission with high probability. An *unused* link is not employed by routing protocols for reasons like bad link quality estimate or absence of a link from the routing table due to strong table-size restrictions.

the fate of future transmission over that link with high probability (using a very limited delivery history of that link). We use Conditional Packet Delivery Functions ($CPDF(n)$) [LCL07] to predict the success probability of the next transmission. We defer a more detailed discussion on $CPDF(n)$ and the thresholds chosen in Table 3.1.1 for defining bursty links to Section 3.4.2. Any non-bursty intermediate link is *independent*.

The *unused* category refers to all the links in the network that, for some reason, are not used by the routing protocol for packet forwarding. The reason could be a smaller table size, the inefficiency of a link estimator or the routing protocol, or the resulting poor estimates of that link.

3.1.2 Requirements

The requirements and challenges of estimating intermediate links are substantially different from conventional link estimation discussed in Section 2.1. For example, long-term packet reception rates – otherwise the key link quality metric – of intermediate links do not suffice as a metric. Rather, we are interested in the following three pieces of information:

- *Whether or not packet delivery on an intermediate link is correlated to its recent delivery history*, i.e., if the link is bursty or independent? It is important because any transmission attempt over an independent link could be compared to mere gambling with an unknown chance of success.
- *How long a bursty link remains reliable for transmission*, i.e., what is the length of successful transmission bursts? It is important because a bursty link that only momentarily becomes reliable for transmission triggers frequent switching among links, degrading the overall routing performance.
- *When a bursty link has a reliable or unreliable transmission period*? We need to pinpoint exactly when a reliable/unreliable transmission sequence occurs over bursty links to ensure their effective utilization.

None of these three pieces of information, which we consider keys to profitably using intermediate links for routing, are provided by existing link estimators.

3.1.3 Major Contributions

The definition of appropriate metrics for estimating intermediate links and the design of a link estimator based on these metrics are the main contributions of this chapter and our major departure from the existing work. The contributions are summarized as follows:

- We introduce MAC_3 as a metric to estimate the burstiness of links based on recent delivery traces. MAC_3 extends the established CPDF [LCL07] by calculating a moving average over the results of CPDF (*Moving Average CPDF*).
- We define EFT as a metric to estimate the duration for which a bursty link remains reliable for transmission. We also show that EFT is strongly correlated to MAC_3.
- Based on these two metrics, we introduce BLE, derive requisite parameters for its usage, and evaluate its efficacy in estimating intermediate links. Our results indicate that BLE identifies bursty links in the network with high accuracy, hence paving the way for including such links in the routing infrastructure.

3.2 Related Work

Capturing link dynamics at different time scales and characterizing link burstiness have been the focus of many recent studies. We can divide prominent related efforts into three main categories.

3.2.1 Measuring Link Burstiness

In their seminal study on quantifying the extent and characteristics of bursty links, Srinivasan *et. al.* [SKAL08] define a factor β that measures the burstiness of a wireless link. β is calculated by using $CPDF(n)$ [LCL07, SKAL08] that determines the success probability of the next transmission after n consecutive successes or failures. Hence, β is used to differentiate between bursty links with long bursts of successes or failures and links with statistically independent packet losses – with perfectly bursty and completely independent links marking the opposite ends of the spectrum.

Although β is a very useful metric to measure link burstiness, it is primarily used to characterize link burstiness based on existing traces rather than online assessment. It is statistically very complex to compute and requires a long delivery history (e.g., 10,000 packets) to accurately measure link burstiness. Our evaluation in Section 3.4.3 also reveals that calculating β over short history sizes, a fundamental requirement for online assessment, results in fluctuating and error-prone results.

3.2.2 Short Term Link Estimation (STLE)

In our preliminary work on wireless link dynamics [BLKW08, ALWB08, Bec07], we introduced the packet snooping based concept of STLE to analyze the impact of the recent transmission success and failure rate on the future quality of a link at fine-grained time scales. However, although STLE is concerned with link estimation, we argue that the proposed mechanism only provides link discovery: STLE only tells whether or not a link becomes temporarily available but does not provide an estimation for how long this will be the case [Gra10]. Furthermore, no difference is made between recurring bursty links and accidental successful deliveries. These characteristics cause STLE to repeatedly select a link even if packet transmissions over that link frequently failed in the previous attempts. This further impedes its usability in real-world networks.

Approaches such as Solicitation-based forwarding (SOFA [LKC06]) remove the need for long-term link estimation and test link availability by sending a short hand-shake packet as a probe before sending any data packets. However, our evaluation in Section 3.4.2 shows that a successful hand-shake should not be taken as a success guarantee for subsequent data transmissions and indicates a need for more sophisticated models.

3.2.3 Long Term Link Estimation (LTLE)

This is the traditional link estimation mechanism employed by the majority of current multihop wireless routing protocols [GFJ+09, FRZ+05]. It is based on window mean exponential weighted moving averages (WMEWMA) of link PRRs or ETX [DCABM05]. Although this metric is highly accurate and has a small settling time for good and bad links, i.e., with PRRs close to 0% and 100%, it does not perform well for links of intermediate quality [WTC03] – also indicated by our results in Section 3.5.1. Hence, such link estimation mechanism cannot be used for estimating intermediate links at short time scales. However, these links often offer the highest routing progress [CWK+05b] suggesting the need for more precise estimation methods.

The assumption underlying the majority of existing link estimation concepts is that packet loss events over a link are independent from each other (i.e., they follow a Bernoulli distribution). This assumption has been challenged before in research [CWPE05, SDTL06b]. The analysis of our data in Section 3.4 supports the hypothesis that the assumption of independent packet losses is not appropriate at the fine-grained time-scales dealt with in this dissertation.

Table 3.2 highlights the main operational differences between existing link estimation mechanisms and our proposed solution. We use 4BLE [FGJL07] and SOFA as reference implementations for our comparison.

3.2.4 Estimating Multiple Link Properties

Both STLE and LTLE mechanisms estimate the quality of a link based on a single link property, such as PRR, ETX or RSSI. The argument used to develop link

Property	4BLE	SOFA	BLE
Passive overhearing	×	✓	✓
Table management	✓	×	✓
Link history	✓	×	✓
Handshaking	×	✓	×
Estimation metric	✓	×	✓
Unicast estimates	✓	×	✓

Table 3.2 Operational differences between link estimators. Handshaking refers to link level connection establishment before data transfer. Unicast estimates specify the ability of a link estimator to monitor data traffic for link estimation purposes.

estimators based on multiple link properties [SKB10,REWT11,KS06] is that a single property is not sufficient to approximate the presumed future behavior of a link.

F-LQE [SKB10] combines four link properties, namely packet delivery, link asymmetry, stability and channel quality into a single hybrid estimate calculated using fuzzy logic [NW05,Har00]. A fuzzy subset [Tan96] of good links is defined using these four properties. The F-LQE link quality indicates the membership of a particular link with this fuzzy subset. Although the authors demonstrate the superiority of F-LQE over existing LTLE mechanisms, it does not meet our design goals of estimating intermediate links. This is because it complements the existing LTLE mechanisms by combining four properties of links that do not reveal the key characteristics of intermediate links such as link burstiness.

Renner *et. al.* [REWT11] argue against the use of a *single-value* metric for expressing the quality of a wireless link. This is because a single-value metric only presents a snap-shot of a particular link property at any particular instance of time. It is unable to express the variation trend both in long- and short-term behavior of the link in the past. They propose a Holistic Packet Statistics (HoPS) concept, which provides detailed information about static and dynamic behavior of a link using four distinct link quality values namely short-term estimation, long-term estimation, absolute deviation and trend. HoPS achieves higher fidelity than existing link estimation techniques and can improve decision making at higher layer protocols such as routing and topology management. However, the short-term link estimates in HoPS are based on PRR and thus they do not provide the required information that we believe is mandatory for exploiting intermediate links.

3.3 Problem Analysis

To provide a clear motivation for our work as well as a separation from the previously mentioned related work, we now define our problem space and the requirements for a solution. First, we motivate the need for employing a new link estimator in networks where LTLE mechanisms are prevalent. Second, we explain the basic concept of our approach by considering a simple example. Based on this, we highlight the key requirements of a link estimator for incorporating bursty links in the routing process.

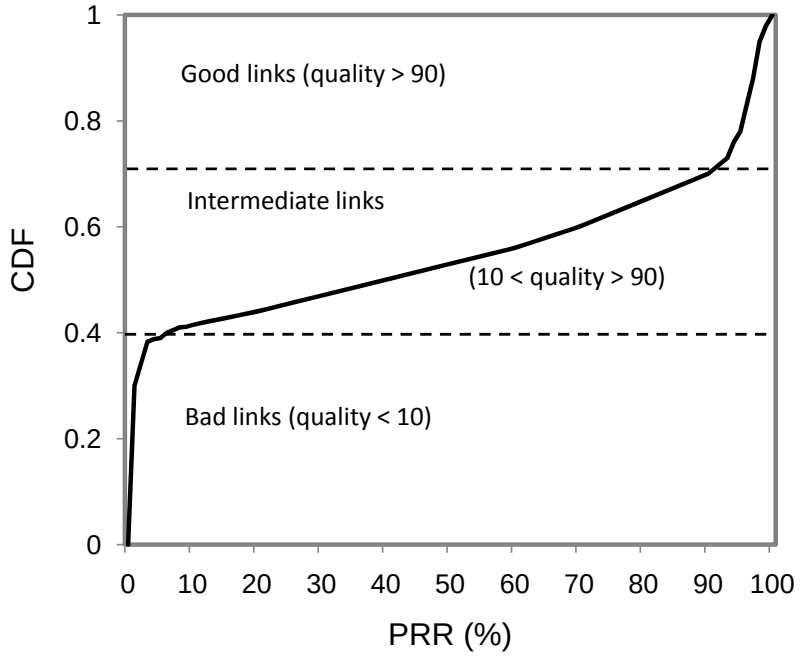


Figure 3.1 The cumulative distribution of different types of links in a wireless network. A considerable amount of links show intermediate to bad quality. The graph shows data from an indoor grid-like deployment of 36 TelosB motes (cf. Section 3.4.1).

3.3.1 The Need to Utilize Bursty Links

With regard to the characterization of links in [SKAL08], LTLE mechanisms typically utilize only good to perfect links with a PRR $\geq 90\%$. However, most links in wireless networks exhibit worse PRRs and are thus excluded from routing decisions (cf. Figure 3.1). Although this approach results in a stable and a clear-cut routing topology, it results in heavy utilization of the selected links that typically offer a little routing progress on each hop. In LTLE, a trade-off is thus made between the high cumulative resource consumption of series of short range links and the ease of utilizing only a fraction of the existing links. In contrast to this, bursty long-range links offer high routing progress with only one transmission but need to be included in the routing process. For example, Figure 3.2 compares the probabilities of finding an intermediate link or a good link when the distance between sender and receiver nodes increases. It clearly shows that the probability of finding an intermediate link is higher in particular at longer distances. However, the utility of these links depends upon an accurate online estimation, which is not possible using today's link estimators.

In multihop wireless networks networks, such as sensornets and meshnets, the networking hardware is the most dominant consumer of energy. The amount of energy consumed by the networking hardware is directly proportional to the number of transmissions required by a packet to reach its destination. By utilizing bursty links with significantly better routing progress (i.e., less number of hops traversed) [BM05a, ALL+09], the number of transmissions and thus, the amount of energy consumed can be reduced.

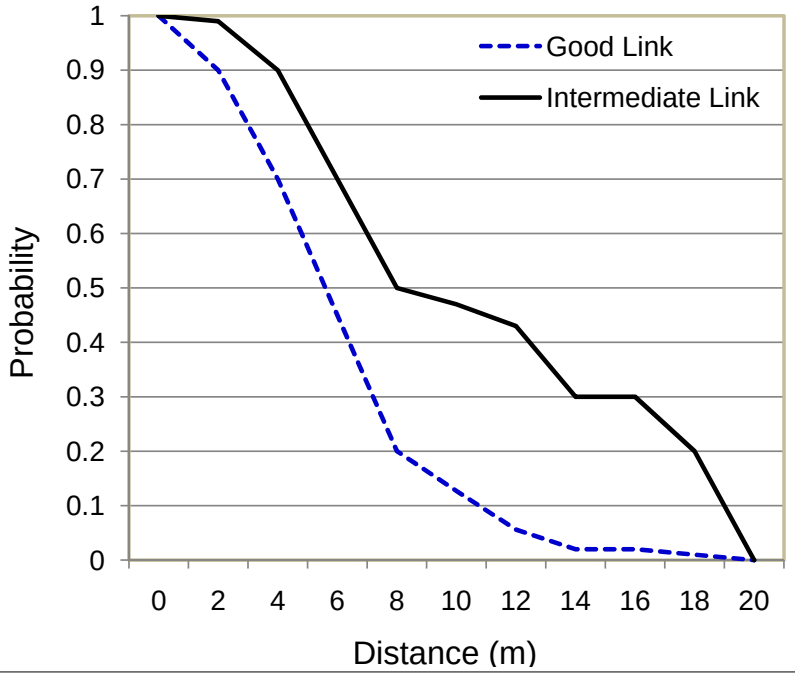


Figure 3.2 Distance vs. probability of finding a particular link category. The probability of finding an intermediate link is higher when the distance between nodes increases (cf. Section 3.4.1 for data set).

3.3.2 Basic Concept

We construct a simple example to understand the requirements of estimating intermediate links. Consider a linear topology of nodes A , B, C, and D as depicted in Figure 3.3. Node A can reach node D via multiple paths such as $A \rightarrow B \rightarrow C \rightarrow D$, $A \rightarrow C \rightarrow D$ and $A \rightarrow D$. Using ETX as link metric, indicated by labeled-edges in Figure 3.3, a traditional link estimation and routing approach will always use $A \rightarrow B \rightarrow C \rightarrow D$ as the only path between nodes A and D. This approach is pessimistic in its link selection because it does not consider the opportunities that may appear during the course of consecutive transmissions on other links, such as $A \rightarrow C$ or even $A \rightarrow D$, by persisting with stable links. Hence, this approach is willing to concede performance penalties to avoid sudden changes in link estimates and routing paths.

An optimistic approach, on the other hand, will speculate the fate of transmissions on long-range intermediate links (e.g., $A \rightarrow C$ and $A \rightarrow D$). This is because if the transmission over such a link is successful, it can potentially reduce the total number of transmissions required by packets from node A to node D. One such optimistic yet expensive approach is ExOR (cf. Section 2.2.2.2), which lets a broadcast packet to be received by a subset of nodes. These nodes then decide on a candidate that forwards the received packet. For example, if both nodes B and C receive the transmission from node A, node C shall forward the packet because it is closer to node D. However, creating such an understanding among node subsets requires maintaining link-state information at each node in the network. Hence, this approach is both expensive and less scalable.

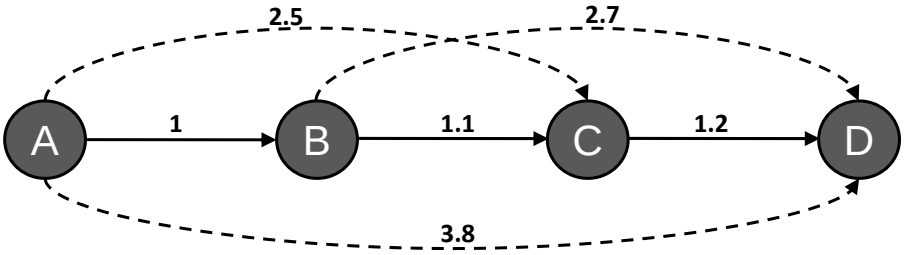


Figure 3.3 Explaining the concept of utilizing intermediate links with a simple example. The labels show the corresponding ETX of the edge.

Another approach is to first seek for an intermediate link, and if the transmission over this link fails, deliver the packet over the traditional path. This dissertation argues in the favor of this approach for two reasons:

- It is simple and based on unicast communication.
- It is distance vector based and therefore, unlike ExOR, does not require each node to maintain the link-state of the whole network.

However, exploiting an intermediate link for transmission without assessing link conditions at the time of transmission confronts one of the following two cases.

- If the intermediate link is *independent* then the fate of future transmissions over this link cannot be predicted. Hence, any speculative transmission attempt over this link will reciprocate to its overall ETX measured by a traditional link estimator.In other words, a 40% link is expected to deliver only 40% of the packets successfully. Hence, using such a link for packet transmission is not desired.
- if the packet loss events over an intermediate link are correlated, i.e., the link is bursty, then the fate of a future transmission depends upon the time when the transmission attempt is made. As a matter of luck, a 40% link may deliver up to 80% of the packets without any retransmissions, if these attempts are made during the reliable delivery period of that link. In the contrary case, a 40% link may deliver less than 10% of the packets.

This dissertation tries to substantiate this luck by enabling the nodes to attempt a transmission over an intermediate link only if it is bursty and precisely at the time when it is showing good transmission characteristics. Hence, in the case of Figure 3.3, if node A possesses this knowledge, it can utilize links $A \rightarrow C$ and $A \rightarrow D$. Estimating link burstiness during runtime and discovering reliable transmission periods over an intermediate link is our main goal in this chapter.

3.3.3 Design Goals

The design of a link estimator that reliably reflects the state of a given link has to fulfill multiple requirements.

Characteristic	Mirage	ComSys
Motes	Micaz	TelosB
Radio chip	CC2420	CC2420
Environment	Indoor	Indoor
Area	160' x 40'	45' x 45'
Tx. power	0 dbm	-25 dbm
Inter packet interval	10 ms	6 ms
Burst length	100000	3000
Channel	26, 11	26

Table 3.3 Transmission characteristics for Mirage and ComSys datasets.

- Appropriate metrics need to be derived as key building blocks of the estimator. Such metrics must timely estimate the current link quality based on a very short transmission history, in order to adapt to the rapidly changing reliable transmission periods of bursty links. Additionally, the predicted link quality must accurately and reliably reflect the actual link quality, i.e., the estimation error needs to be small and stable.

- Building upon such metrics, a link estimator must efficiently utilize the given information to select beneficial links for routing. This requires appropriate neighbor table management policies that select those links for routing – among all the available links – which allow for the best routing progress.

- The link estimator mechanisms should be lightweight and resource sensitive in terms of computation, storage, and communication. For example, frequently broadcasting beacons to estimate a link on a short-term basis is prohibitive because it consumes significant amount of energy and bandwidth, the two most critical resources in sensornets.

3.4 Deriving Metrics for Bursty Links

Based on the properties specified in the previous section, this section defines and evaluates two metrics, MAC_3 and EFT, that (1) identify bursty links in a network and (2) estimate the length of successful transmission bursts. These metrics subsequently lay the foundation for BLE. In the following, we first provide detailed information on the particular data set used in the remainder of this study.

3.4.1 Data Set and Experimental Model

The design of our link quality metrics as well as the resulting BLE are based on widely used empirical data rather than a theoretical model. We strongly believe that empirical observations from multiple real world scenarios are important both for developing metrics and evaluating the efficacy of the concepts presented in this dissertation. Although attempts have been made previously, it is difficult to capture transmission fluctuations and the dynamics revealed by intermediate links only using a theoretical model [SKAL08, LCL07, KCPnC09].

The evaluation results presented in this chapter are based on two different datasets. First, the SING mesh data-set [Dat09] compiled at Stanford University and used in many recent state-of-the-art studies [SKAL08, SJC+10, Sri10] on wireless link dynamics. It is a comprehensive data set collected from multiple IEEE 802.11 and IEEE 802.15.4 testbeds[1] including both packet and byte level radios such as Texas Instrument's CC2420 [Ins07] and CC1000 [Chi02]. Second, the ComSys data set from Becher *et. al.* [BLKW08], collected from an indoor deployment of 36 TelosB [PSC05] motes. Unless otherwise noted, our evaluation shows results from Mirage testbed [CBA+05] – a 100 node MicaZ [SA08] testbed at Intel Research Berkeley. Specifically, the data comprises traces of transmissions on IEEE 802.15.4 channel 26 at a transmission power level of 0 dBm. Out of all available links, we only include intermediate links in our analysis and comparison. This is because good links would measure highly in our metrics and would thus improve our results. However, these links are not the focus of our work.

Table 3.4 lists the important characteristics of our data set. The *burst length* indicates the number of packets broadcast by one node before passing on this duty to the next sender in the network.

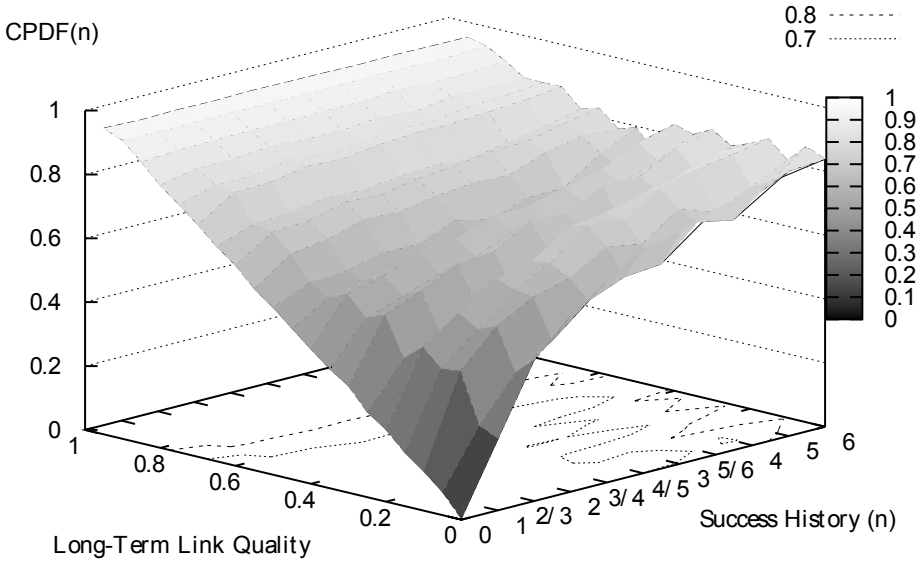
3.4.2 Predicting Transmission Success from a Short History

Before introducing our metrics MAC_3 and EFT we motivate our goals and approach with a case study. We address the question, whether a short history of successful transmissions is sufficient to predict with a high probability that the next transmission on this link will be successful, too. Our goal is to find a minimum threshold that is sufficient to predict the fate of future transmissions.

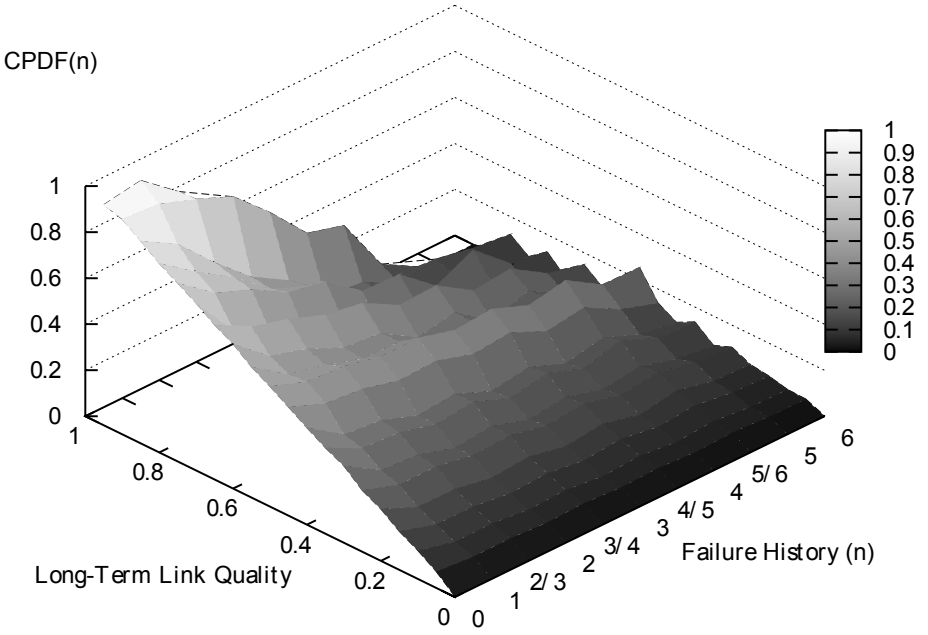
Figure 3.4(a) depicts the conditional probability of a successful packet transmission based on the average long-term link quality (i.e., PRR) and a short-term history of a link. It shows that for a link with a long term quality greater than 60%, even a single or two successful transmissions over that link raise the success probability of the next transmission to 90%. Similarly, it shows that for any link, regardless of its long term link quality, the probability of a future successful transmission is greater than 90% if the last three packets over that link were sent successfully. Figure 3.4(b) depicts the probability of a successful packet transmission based on the average long-term link quality and a short-term history of consecutively *failed* packet transmissions. It indicates that after one or two consecutive losses any link should be temporarily considered unreliable.

Overall, these results indicate that a short-term history of three packets over a link is sufficient to determine with a high probability whether the next transmission will be successful or not. Hence, from this case study, we derive a history of 3 packets as a suitable threshold for discovering reliable periods of transmission over bursty links. A smaller threshold value, such as 1 or 2 packets, is risky as it does not guarantee a successful delivery over an intermediate link. Similarly, using a larger threshold value, such as > 3 packets, does not significantly increase the success probability of future transmissions while impeding the prediction process.

[1] MoteLab [WASW05], Mirage [CBA+05], and SWAN [Sta] testbeds. Please visit `http://sing.stanford.edu/srikank/datasets.html` and the websites of each testbed for further information, e.g., topology, connectivity etc.

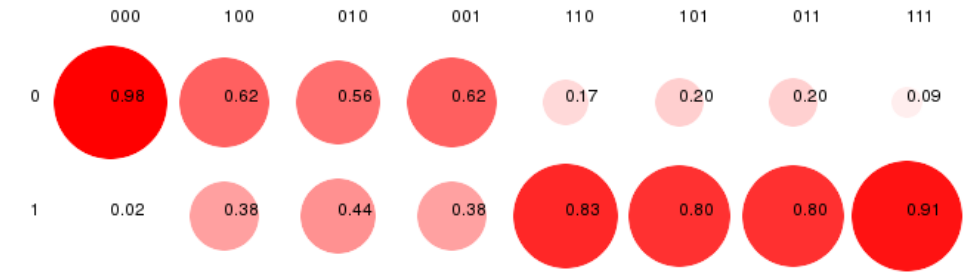


(a) Influence of the recent transmission success rate on the success probability of the next transmission.



(b) Influence of the recent transmission failure rate on the success probability of the next transmission.

Figure 3.4 Measuring the impact of recent transmission success or failure over a link on the next transmission over that link. A label of k/n stands for k successes during the last n transmissions, and n is a shorthand for n/n. CPDF(n) is the probability that the next transmission is successful. Long term link quality reflects the PRR, calculated over the whole link trace.



(a) $h = 3$

(b) $h = 4$

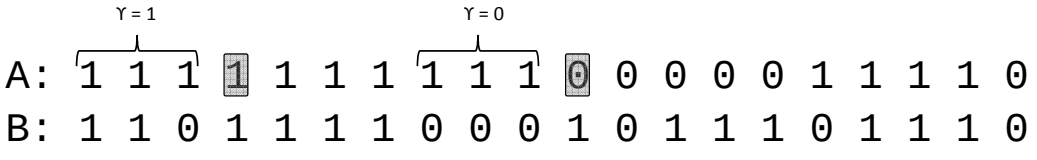
Figure 3.5 Conditional packet reception probability of two distinct history sizes, (a) $h = 3$ and (b) $h = 4$. The probability is shown for all possible combinations of packet loss and reception events (first row) for a particular history size. Symbols 0 and 1 represent packet loss or reception event (first column), respectively. The radius of the circle scales with the probability of a particular event [Bec07].

Figure 3.5 precisely illustrates the packet loss and success probability of the next transmission for all possible combinations of packet loss and success events using two distinct history sizes (i.e., $h = 3$ and $h = 4$). We can clearly see that $h = 3$ suffices for predicting the success probability of next transmission.

In the following, we introduce two metrics MAC_3 and EFT that determine the success probability of future transmissions on a per link granularity, hence allowing us to reflect spatial properties of link dynamics.

3.4.3 Online Estimation of Link Burstiness

Estimating the burstiness of a link is mandatory to determine whether or not an intermediate link is beneficial to the overall routing performance. The key challenge is to clearly distinguish intermediate links with correlated packet losses from those with independent losses. However, unlike offline measurement mechanisms like β, we are not interested in how close a link is to an ideal bursty link with one long burst of either successes or failures. Our goal to predict link burstiness at runtime strongly influences the definition of burstiness and the timescale of our prediction. In this context, we define a link to be bursty as long as we can *recurrently* predict the fate of only the next transmission over a link with high probability. This is why we introduce a new metric that monitors a link for a limited transmission history and expresses if the occurrence of a successful transmission burst over a particular link is a mere coincidence or if it is a reoccurring trend. This information is important to determine if a link is beneficial for routing purposes.



Variables	Link A	Link B
ρ	10	4
ϒ = 1	8	1
ϒ = 0	2	3
CPDF(3)	8/10 = **0.80**	1/4 = **0.25**

Figure 3.6 Calculating CPDF(3) for two contrasting links. Link A has a higher CPDF(3) and is more suitable for routing.

3.4.3.1 MAC$_3$

Our online metric Moving Average $CPDF$ (MAC$_3$) is based on $CPDF(n)$ which calculates the probability of one successful transmission following n previously successful transmissions. Based on the results in the initial case study in Section 3.4.2, we compute $CPDF(3)$ over the recent history h of length m of a link:

$$CPDF(3) = \frac{\sum_{i=1}^{m} \gamma_i}{|\rho|} \tag{3.1}$$

ρ defines the total occurrences of three consecutive successful transmissions in the history. For example, in a transmission history 111001, there is only a single valid occurrence of three consecutive successful transmissions. γ is a boolean function defined as follows:

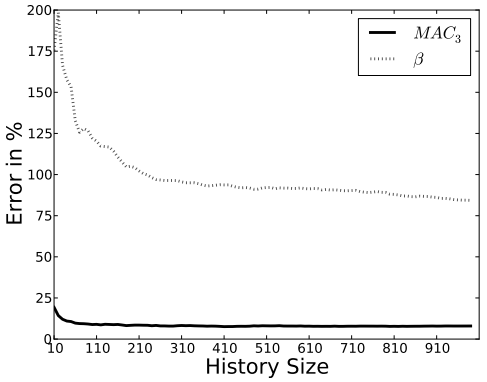
$$\gamma_i = \begin{cases} 1 & \text{if } h_i = 1 \wedge h_{i+1} = 1 \wedge h_{i+2} = 1 \wedge h_{i+3} = 1 \\ 0 & \text{otherwise} \end{cases} \tag{3.2}$$

h_i represents the corresponding entry for i^{th} packet in the link history, i.e., 1 if it was received and 0 otherwise. We define MAC$_3$ as an exponentially weighted moving average over the values of CPDF(3) with a weight of α that controls the history of MAC$_3$.

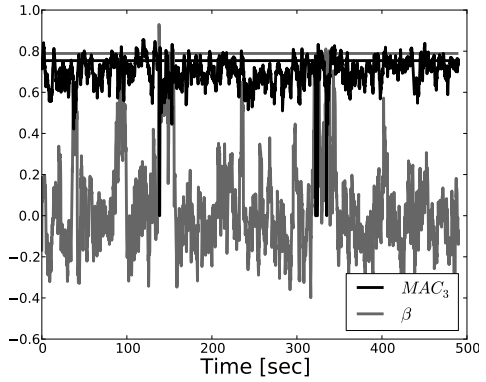
$$MAC_3 \;=\; (\alpha)MAC_{3(old)} + (1-\alpha)CPDF(3)$$

3.4.3.2 Example

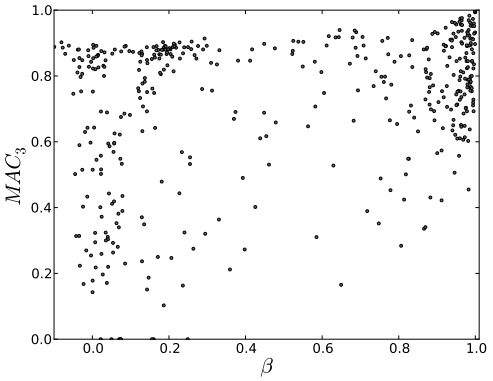
We present a simple example to explain how MAC$_3$ is calculated using the link traces in Figure 3.6. Calculating CPDF(3) requires looking at each instance of three consecutive successful transmissions in the trace and determining if the next



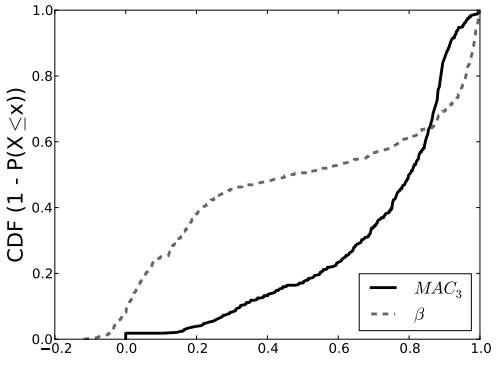
(a) Settling Time: MAC_3 shows a faster convergence towards its base value over the history size and achieves a smaller estimation error (7%) than β. Based on data of the Mirage testbed.



(b) MAC_3 generates more accurate and more stable results over time than β. The straight lines show the base values of both metrics over the entire transmission trace.



(a) MAC_3 reveals that many links have a high probability for a further successful transmission after three consecutive deliveries even though their β is very low.



(b) Cumulative distribution of intermediate links: The majority of intermediate links is bursty ($MAC_3 > 0.7$), offering useful transmission opportunities.

Figure 3.8 Comparing MAC_3 and β as a link burstiness metric for runtime link estimation. We use a smaller version of β for online link assessment. Our version of β does not enforce a confidence interval of 95% for its data points.

transmission is successful or not: A successful transmission means $\gamma = 1$. For example, in Figure 3.6, link A has eight occurrences of $\gamma = 1$ and two occurrences of $\gamma = 0$. CPDF(3) is calculated by dividing the total number of $\gamma = 1$ occurrences with ρ. MAC_3 is a moving average over the values of CPDF(3) as shown in Equation 3.3.

3.4.3.3 Results

To evaluate MAC_3 we compare it with the β factor [SKAL08] because, (1) it is the only metric available that measures link burstiness and, (2) it enables a better understanding of the effectiveness of MAC_3 as a runtime metric. However, this comparison, by any means, does not attempt to undermine the usefulness of β as it was not developed for runtime measurements.

Figure 3.7(a) illustrates the estimation error of MAC_3 and β over history sizes ranging from 10 up to 1000 packets. The estimation error is the difference between the estimated value of either metric when applied to a history of certian size (plotted on the x-axis) and the value when applied to the whole transmission trace (i.e., base value). The base value of β is calculated according to the procedure described in [SKAL08]: A $CPDF(n)$ for a certain n is only considered in β calculations if it has at least 100 data points to achieve a 95% confidence interval of [p-0.1, p+0.1]. Whereas for calculating β over a shorter transmission history, we do not enforce the condition of 100 data points. This is because, (1) it is simply not possible to collect 100 data points in a shorter transmission trace, and (2) we want to investigate if this restricted version of β provides accurate estimates and can be used for runtime estimation of link burstiness. The figure indicates that our online metric MAC_3 rapidly converges to an error of 7% within a history size of less than 100 packets. In contrast, β shows a significantly slower initial convergence phase and is not able to achieve an error smaller than 83% even with a history size of 1000 packets.

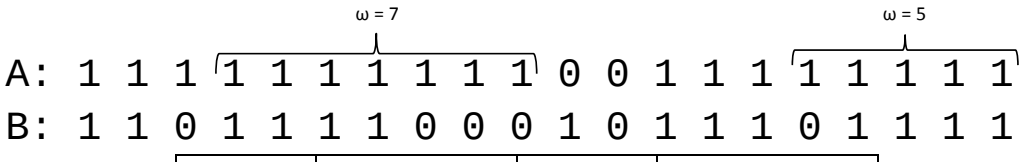
Moreover, β is not able to provide stable results for small history sizes as shown in Figure 3.7(b). Given a concrete history size, β generates severe fluctuations in its output over time when applied to an entire transmission trace of a particular link. The estimated values of MAC_3 on the other hand exhibit considerably smaller differences. In addition, the results of β again strongly deviate from the base value calculated over the whole trace (straight gray line) while the estimates of MAC_3 oscillate around its actual base value (straight black line). Overall, these results show the efficiency of MAC_3 as a runtime metric: It is stable for short history sizes.

Next we show that MAC_3, in contrast to β, captures the short term behavior of a link. Figure 3.8(a) shows that many links with $MAC_3 > 80\%$ have low β values. It means that on such links the probability of a successful transmission after three consecutive deliveries is greater than 80%, but the use of β as a link metric will not let a routing protocol select this link. After evaluating the effectiveness of MAC_3, we need to analyze what proportion of the available intermediate links are actually useful for routing. Figure 3.8(b) shows the cumulative distribution function of MAC_3 and β for all the intermediate links in the Mirage testbed. We can clearly observe that the majority of these links have a very high MAC_3. As a result, MAC_3 unlocks the formerly wasted potential of those links and enriches the routing process with a multitude of new routing opportunities.

Concluding, MAC_3 is a lightweight metric for estimating link burstiness during runtime. Our results in Section 4.4.3.3 demonstrate that, when used as metric to estimate link burstiness, MAC_3 accurately identifies bursty links in the network.

3.4.4 Estimating Burst Lengths

In addition to identifying whether or not a link is bursty, a second metric for estimating the length of bursts is required. To illustrate why, assume a bursty link with a steady rate of bursts covering four successful transmissions each before becoming unreliable again. Such a link exhibits a high $CPDF(3)$ value, causing MAC_3 to correctly identify it as bursty. However, if selected for transmission, this link allows for only one more successful transmission per burst, hence rendering it barely suitable for routing.



Variables	Link A	Variables	Link B
ω = 7	1	ω = 0	1
ω = 5	1	ω = 1	2
η	2	η	3
FPDF(3)	(7+5)/2 = **6**	**FPDF(3)**	(2x1+0)/3= **0.67**

Figure 3.9 Calculating FPDF(3) for two contrasting links. Link A has a higher FPDF(3) and is more suitable for routing..

3.4.4.1 EFT

In order to identify links with relatively longer transmission bursts, we introduce a new metric named EFT. It is based on FPDF(n) (Future Packet Delivery Function) that calculates the number ω of successful future transmissions after n successful packet deliveries:

$$FPDF(3) = \frac{\sum_{i=1}^{m} \omega_i}{|\eta|} \tag{3.3}$$

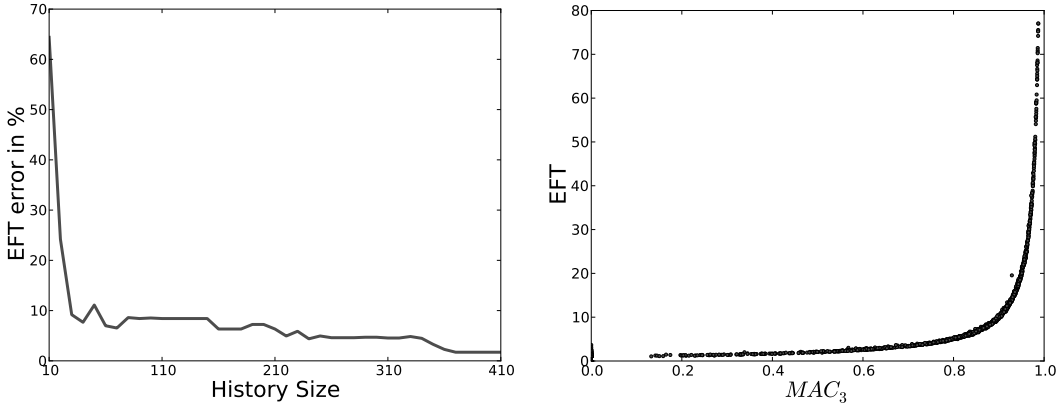
η defines the total number of transmission bursts with a minimum length of three. This metric thus predicts the length of bursts and allows the link estimator to identify bursts of relevant size. Just like MAC_3, EFT uses an averaging moving window to traverse a transmission history:

$$EFT \;\; = \;\; (\alpha)EFT_{old} + (1-\alpha)FPDF(3)$$

3.4.4.2 Example

We consider a simple example to explain how EFT is calculated using the link traces in Figure 3.9. Calculating FPDF(3) requires looking at each transmission burst with a minimum length of three and then counting the remaining successful transmissions in that burst. For example, in Figure 3.9, link A has two such bursts with $\omega = 7$ and $\omega = 5$. After determining all the ω values in a link trace, FPDF(3) is calculated by averaging ω using Equation 3.3. EFT is a simple moving average over the values of FPDF(3).

As FPDF metric determines the length of a burst, its calculation procedure is slightly different from CPDF in that we only calculate one ω per transmission burst and then move on to the next burst. Whereas in CPDF, γ is calculated for each occurrence of three consecutive transmissions, i.e., multiple γ values can be calculated during a single transmission burst.



(a) Influence of the history size on the convergence of EFT. Similar to MAC_3, EFT quickly converges to a very small error from its base value.

(b) Correlation between EFT and MAC_3. EFT helps in settling the thresholds for MAC_3: An average burst length of size 10 requires a MAC_3 value of at least 0.7.

Figure 3.10 EFT and MAC_3 as link quality metrics. EFT has a smaller convergence time and shows a strong correlation with MAC_3.

3.4.4.3 Results

Our evaluation of EFT shows a similar trend as MAC_3: It has a very small settling time (cf. Figure 3.10(a)). It converges to within 10% error at a history size of approximately 100 packets. Figure 3.10(b) indicates this strong correlation between EFT and MAC_3. For values of MAC_3 in the range of 0.1 to 0.7, EFT predicts burst lengths not longer than five packets. However, when MAC_3 exceeds 0.7, the estimated burst lengths increase significantly. As a result, we derive a MAC_3 threshold of 0.7 for a link to be considered useful for routing.

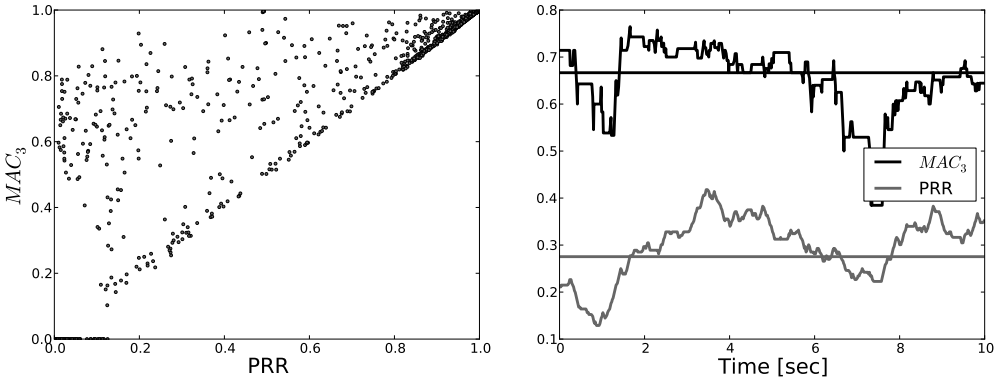
3.5 The Bursty Link Estimator

BLE employs a packet snooping based link estimation mechanism [WTC03,FRZ+05]. It is not supposed to work independently: It is an additional component of the routing infrastructure that enables a fine grained estimation of intermediate links and allows for such links to be included in the routing process. In this section, we first discuss why PRR is not a suitable metric for intermediate links and propose a combination of MAC_3 and EFT to be used as link quality metrics for BLE. We then provide further details about the information maintained in BLE's table. Finally, we conclude this section by evaluating BLE.

3.5.1 Link Quality Metric

PRR (or ETX: the reciprocal of PRR) is commonly used as a link metric in current link estimators. The basic technique is to calculate weighted moving averages of PRR over a long time period.

Similar to β, PRR does not fulfill the desired properties of a metric for our envisioned link estimator. For example, it is unable to capture short term dynamics exposed by



(a) Many links with low PRR exhibit a high success probability after 3 successful deliveries. All links of Mirage testbed are plotted in this graph.

(b) A 28% link shows a high MAC_3 and stable progress over time. PRR will never include such a link in neighbor tables.

Figure 3.11 Comprison of MAC_3 with PRR: MAC_3 identifies potentially valuable communication links in the network with bursty transmission characteristics. Compared to PRR, it assigns higher estimates to links of different qualities.

bursty links of intermediate quality. Figure 3.11(a) highlights this fact: Many links with a very high MAC_3 have very low PRRs. It means that over a long time scale these links have bad reception rates. However, when observing a limited transmission history, i.e., the last three packets, it is possible to predict the success of future transmissions with high probability. Hence, when using PRR as a link estimation metric, these links cannot be be utilized even though reliable transmission periods occur frequently over these links. Figure 3.11(a) also shows that, when compared to their values of PRR, all links get the same or higher value of MAC_3. Thus, MAC_3 exhibit the properties of PRR and can be considered as a suitable candidate to replace PRR as link estimation metric.

Similarly, Figure 3.11(b) supports this argument by comparing PRR and MAC_3 over time. It shows that although MAC_3 indicates a high probability of successful delivery, PRR is unable to capture this reliable transmission period of a link. Hence, the use of PRR prohibits the use of bursty links that offer useful transmission opportunities at shorter time scales.

Finally, Figure 3.12 depicts the correlation between β and PRR. We can clearly see that even bad links (i.e., PRR < 10 %) can posses high β values. This emphasizes on the importance of EFT metric: β is independent of the length of successful transmission burst. Hence, even bad links, such as the ones with a rare occurrence of successful transmission burst, can attain a high β value.

In BLE, we use a hybrid metric that is based on the product of MAC_3 and EFT. Both these metrics are calculated by applying a sliding window over the packet delivery history of size h for each link in the table. Since maintaining the link history is an expensive memory operation and impacts the scalability, it is important to choose the threshold h appropriately as discussed in Section 3.5.3.1.

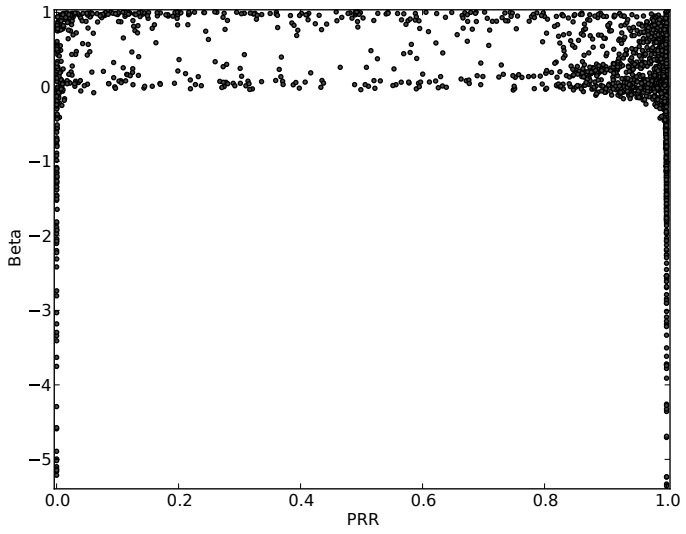


Figure 3.12 PRR vs β: Many links with low PRR values (e.g., $< 10\%$) can attain high β values. β is independent of the length of the transmission burst.

3.5.2 Table Management

BLE follows the basic table management algorithm outlined by Woo *et. al.* [WTC03] and used by the majority of current link estimators [FRZ+05, FGJL07]. We deviate from the established concept in terms of (1) link selection as BLE only estimates *unused* links, and (2) different ingredients for the link insertion, eviction, and reinforcement policies. The estimator maintains a small table (e.g., of size 10) of candidate links which holds the following information per link:

- $\mathbf{MAC}_{3in}$: The reception MAC_3 of the link.
- $\mathbf{EFT}_{in}$: The reception EFT of the link.
- $\mathbf{MAC}_{3out}$: The sending MAC_3 of the link.
- $\mathbf{EFT}_{out}$: The sending EFT of the link.
- **Link History**: The packet delivery history of size h. Bit arrays are used with 1 representing a successful delivery and 0 representing a failed transmission.
- **Available**: A flag to determine if the link, with MAC_3 and EFT above certain threshold, is currently available for transmission. Set to 1 if the last three transmissions over the link were successful, and 0 otherwise.
- **Valid**: A flag to determine if the link has a large enough delivery history, and all other table entries are up-to-date.

The table management is concerned with three tasks: Adding links, deleting links and maintaining links in the table. A new link is added to the table upon reception of a packet on a non-resident link and (1) a vacant entry in the table exists, (2) the product of MAC_3 and EFT of a resident link drops below a user-specified threshold,

or (3) an entry expired due to a broken link or an insufficient packet reception rate. Additionally, link maintenance is performed after i received packets. At this point, all entries in the table are recalculated. The value i is a trade-off between the computational overhead and the actuality of BLE.

3.5.3 Evaluation

We have implemented a prototype of BLE in TinyOS for sensornets. Our evaluation of BLE focuses on three factors:

- *Link History Size*: We empirically derive a requisite history size h that shall be maintained by BLE to compute its link metrics.
- *Link Estimation*: We validate that BLE indeed includes bursty links of high quality in the neighbor table.
- *Routing*: We integrate a prototype of BLE with an existing routing protocol and link estimator to assess its potential benefits.

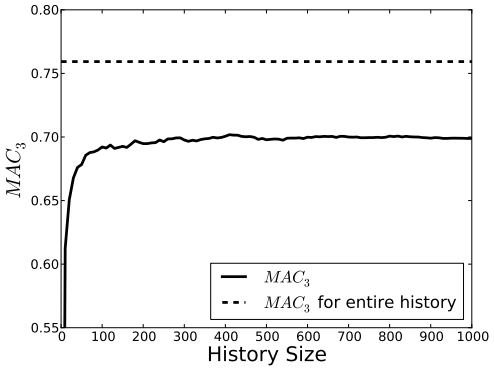
Among these three factors, the *link estimation* constitutes the key in assessing the performance of any link estimation mechanism, as the quality of the link selection process has a significant impact on the overall routing efficiency. Similarly, the primary purpose of link estimation is the selection of beneficial links for routing.

3.5.3.1 Link History Size

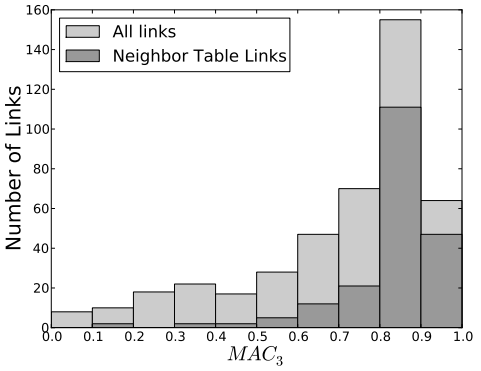
Although determining an appropriate link history to calculate link estimation metrics during runtime is a user-desired accuracy threshold, we derive its value here for completeness and for evaluation purposes. Our goal is to find a requisite history size that balances estimation error and memory consumption. A too small history does not provide enough information to enable BLE to accurately predict the link quality. Conversely, a too large history blocks valuable system resources and potentially does not even improve prediction accuracy. We assume that an estimation error of 10% yields user-acceptable results. Figure 3.7(a), 3.10(a) and 3.13(a) show our results derived from the data set of the Mirage testbed. We can clearly observe that MAC_3 and EFT converge below a 10% error at a history size of approximately 100 packets. Hence, for our evaluation, we derive MAC_3 from CPDF(3) values that correspond to a link history of at least 100 packets.

3.5.3.2 Link Estimation

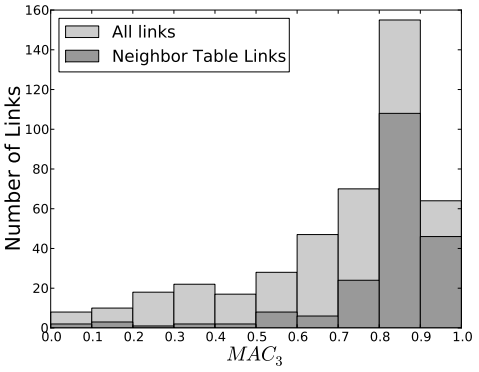
This evaluation aims to confirm that BLE correctly identifies bursty links in the network to provide these links with a high value of MAC_3 for inclusion in the routing process. Figures 3.13(b) and 3.13(c) illustrate the total number of links with a certain estimated quality and the fraction of links that were included in the neighbor table by BLE. We observe that the fraction of selected links increases in conjunction with



(a) Influence of the history size on convergence of MAC_3. It settles to a less than 10% error within a history size of 100 packets.



(b) Number of links with a given estimated quality (light gray); subset of these links that are included in the neighbor table after 1000 transmissions by each node in the network (dark gray).



(c) Number of links with a given estimated quality (light gray); subset of these links that are included in the neighbor table after 2000 transmissions by each node in the network (dark gray).

Figure 3.13 Evaluating BLE: MAC_3 is a suitable link estimation metric because of its small convergence time. Using MAC_3 as a link estimation metric, BLE accurately identifies bursty links in the network and includes them in the neighbor table.

the estimated link quality. The fact that not all links with a high value of MAC_3 are included for routing stems from the criteria of link addition (see Section 3.5.2) and the requirements of a fixed and small table size. As a result, there may exist more suitable links than can be included in the table. Although Figure 3.13(b) and 3.13(c) present instantaneous snapshots of the BLE tables, we observed a similar trend throughout our evaluation.

3.5.3.3 Routing

Although an advanced routing evaluation is deferred until the next chapter, we present initial results here for completeness. We integrated BLE with the standard CTP [GFJ+09] and the 4BLE [FGJL07] shipped with TinyOS. By integrating BLE with CTP, we allow CTP to use long range intermediate links whenever (1) BLE

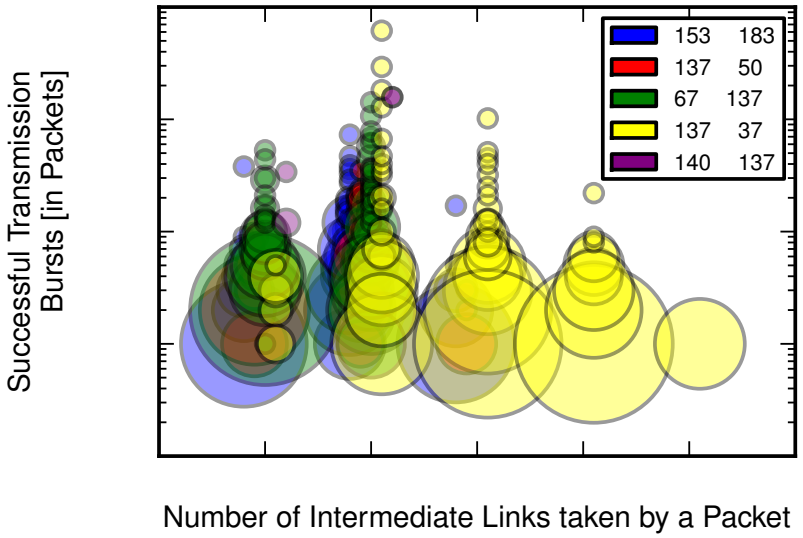


Figure 3.14 The number of bursty links taken by a packet and the burst length on the path from source to destination. A randomly selected set of node-pairs (see legend) is used from MoteLab as senders and collection roots. The radius of the circle shows the number of occurrences of such transmission bursts. Please note the logarithmic y-axis.

declares a bursty link reliable for transmission, (2) the MAC_3 of that link exceeds the predefined threshold, and (3) the declared link offers a shorter routing path than the link currently used by CTP (i.e., by comparing their hop counts to the collection root). We randomly selected 5 node pairs[2] from MoteLab [WASW05] as senders and collection roots. The maximum path length between these node pairs is 5 hops.

We want to analyze the following three factors:

- How many intermediate links are taken by a packet on its path from source to destination (see x-axis in Figure 3.14),
- What is the length of successful transmission bursts over these intermediate links used by CTP (see y-axis).
- How often these successful transmission bursts of a particular length occur on an intermediate link (indicated by the radius of the circles). This factor is important to observe if an intermediate link becomes repeatedly reliable for transmission or if a successful transmission burst over this link is a mere coincidence.

Figure 3.14 shows that BLE enables routing protocols to use the previously ignored class of intermediate links with longer successful transmission bursts. It also shows that a packet takes multiple intermediate links on its way from source to destination. Moreover, we can clearly see that these links become repeatedly reliable for transmission as indicated by the radius of circles. Hence, these results prove the principle feasibility of BLE for routing.

[2]Please visit http://motelab.eecs.harvard.edu/ to see the exact location of the selected node pairs and the overall network topology

3.6 Summary

To achieve better connectivity and a more reliable packet communication, today's link estimators restrict communication to neighbors with constantly high-quality links. These links are identified based on the long-term success rate of a link collected over a time frame in the order of minutes. However, this approach has two major pitfalls. First, neighbors with intermittent connectivity might reach farther into the network. Their use might therefore offer better routing progress and hence reduce the number of transmissions, lower energy usage in the network, and increase throughput. Second, in a sparse network with a low density of nodes, a node might have no high-quality neighbor in its communication range, and therefore requires a mechanism to deal with unstable connectivity.

In order to overcome these limitations of today's link estimators, we presented a bursty link estimator that allows the inclusion of bursty links into the routing process, thereby enabling a better utilization of the existing links in a network. We observed that the traditional metrics, such as β and PRR (or ETX), used to *measure* link burstiness and link quality, respectively, are of limited use in *estimating* intermediate wireless links. In this regard, we presented MAC_3 and EFT as metrics to estimate link burstiness and burst lengths of intermediate links, respectively. Our evaluation on testbeds demonstrates that a link estimator based on these two metrics, i.e., BLE, accurately estimates intermediate links and enables inclusion of bursty links in the neighbor table. BLE does not replace the existing link estimation mechanisms. Rather, it brings additional knowledge to the the table in the form of link burstiness estimates at much shorter time scales with the aim to improve existing mechanisms.

In the next chapter we move the focus of our discussion towards the routing layer. We develop an adaptive routing strategy that uses bursty links for packet forwarding. We also show how BLE can be integrated with existing routing protocols and link estimators.

4

Routing over Bursty Wireless Links

Routing protocols aim at establishing a stable routing topology based on the links suggested by their link estimators. Hence, accurate estimation of link quality is the key to enable efficient routing in multihop wireless networks. In Chapter 3, we presented relevant metrics and a specialized link estimator that estimates intermediate links.

This chapter takes the next step in our pursuit to include intermediate[1] links into the routing process. To this end, we introduce BRE that provides relevant support at the routing layer to utilize such links for packet forwarding over multiple hops. Due to the highly dynamic nature of intermediate links, BRE employs a very cautious approach of forwarding packets over intermediate links. The main purpose of BRE is to enable seamless integration of short-term link estimators, such as STLE and BLE, in the existing routing infrastructure without compromising the stability of existing routing protocols. The idea is to strike an efficient tradeoff between the stability of routing topology and the adaptability of routing paths to the underlying link conditions.

Our evaluation on widely used IEEE 802.15.14 testbeds indicate that BRE achieves an average of 19% and a maximum of 42% reduction in the number of transmissions when compared to the state-of-the-art collection tree protocol. Moreover, we show that BRE is not tied to any specific routing protocol and integrates seamlessly with existing routing protocols and link estimators.

The remainder of this chapter is structured as follows. We motivate the problem space and introduce BRE in Section 4.1. Section 4.2 discusses related work. Section 4.3 presents the basic concept and outlines design goals. The design of BRE and the associated challenges are discussed in Section 4.4. We present our evaluation results in Section 4.5. Finally, Section 4.6 concludes the discussion.

[1]We use the term *intermediate* and *bursty* interchangeably throughout this chapter. Bursty links refer to the intermediate links with high CPDF(n) as shown in Table 3.1.1.

4.1 Introduction

Instability of links and connectivity in low-power sensornets has so far been regarded as a difficult problem that existing routing algorithms try their utmost to avoid. Therefore, since the emergence of sensornets, research has mainly focused on link estimation and routing techniques [FGJL07, FRZ+05, WTC03, GFJ+09] which identify and utilize consistently high quality links for packet forwarding. Links of intermediate quality are typically ignored to ensure routing stability and to attain high end-to-end reliability. In this chapter, we argue that: (1) Bursty links can be used for packet forwarding during their stable transmission periods without affecting the reliability and stability of existing routing protocols, and (2) these links often achieve significantly better routing progress and routing throughput than the long-term links chosen by existing routing protocols.

4.1.1 Significance

A great deal of effort has been invested in evaluating and quantifying the varying and dynamic characteristics of links in wireless networks both analytically and experimentally. To fine-tune protocol parameters operating at different layers of the network stack, these studies have led to the definition of analytical metrics and experimental parameters, such as link burstiness. Our major departure from the existing work is that we neither introduce any new experimental model for wireless links nor define any parameters for fine-tuning protocols. We exploit the existing knowledge on burstiness of wireless links to enhance network performance. Similarly, we investigate the applicability and practicality of bursty links that often offer the highest routing progress, instead of devising mechanisms to bypass them for the sake of stability.

Widespread routing protocols in sensornets, such as BVR [FRZ+05] and CTP [GFJ+09], select links as suggested by their link estimators. In doing so, they limit packet forwarding only to long-term reliable links and forgo a large class of potentially valuable communication links of intermediate quality. In order to utilize these intermediate links in the routing process, we use short-term link estimators, such as STLE and BLE, that capture link dynamics at a high resolution in time. These link estimators identify the periods when bursty links become temporarily reliable or unreliable for transmission. However, while maintaining a stable network topology, we do not replace existing link estimators. We introduce BRE as corresponding routing extensions that enable seamless integration such link estimators with existing routing protocols and link estimators.

4.1.2 Key Features

BRE has four distinguished features:

- It is not a routing protocol but an extension that enriches existing routing approaches to (1) exploit the formerly waisted potential of intermediate links, and (2) enhance routing performance by enabling better utilization of wireless

resources. Owing to the severe resource constrained nature of sensornets, BRE is resource sensitive in terms of processing, energy, memory and, communication bandwidth: it has a less than 1 *kb* of code and data memory footprint.

- BRE owns a highly modular design that transparently integrates with existing routing infrastructure. It is independent of the underlying routing infrastructure and can easily accommodate any routing algorithm that uses PRR based cost metrics for establishing routing paths. The key to such transparent integration lies in ensuring that the communication related behavior of the underlying routing infrastructure is not altered. Section 4.4.1 demonstrates that BRE fulfills this requirement by avoiding any changes in the routing header and in the basic forwarding logic.

- It maintains the end-to-end delivery reliability of traditional routing. This is important because packet forwarding over lossy links introduces a potential risk of high packet loss. To this end, BRE employs a very careful approach to ensure that intermediate links are not overexploited. In the case of packet loss, BRE rapidly falls back to traditional routing for delivering the lost packets. Section 4.5.3.2 empirically proves that BRE maintains the delivery reliability of traditional routing. Moreover, in challenging network conditions, it even improves the delivery reliability by rapidly adapting routing paths to the underlying link conditions.

- BRE finds a suitable tradeoff between the stability and adaptability of routing. Packet forwarding over intermediate links requires a highly adaptable routing approach. However, rapid changes in the underlying routing topology are prone to typical routing pathologies such as loops and stranded nodes in network. Due to this BRE only optimizes link selection locally, i.e., within one-hop vicinity of a sender node, and avoids spreading these local optimizations across the whole network.

4.2 Related Work

The BRE algorithm presented in this chapter is designed according to the lessons learned from experimental studies on bursty wireless links, such as [SKAL08,ABB+04, JSX02]. However, our work does not aim at modeling and developing analytical or experimental understanding of wireless links. Instead, we take a step further and use these experimental models for packet forwarding over bursty links and hence enable better utilization of wireless links.

In recent years, a great deal of effort has been invested in investigating and exploiting link dynamics at the network layer to improve routing performance. As opposed to traditional routing approaches, which only employ a single routing path between two communicating nodes, these approaches leverage *unused* links and paths leading towards a destination. In the following we discuss three such prominent related efforts.

4.2.1 4C: Wireless Link Prediction

4C [LC11] comes closest to BRE both in terms of its goals and how it operates. In fact it uses BRE as its routing strategy [LC11] (it was developed after[2] BRE). In other words, it replaces BLE as BRE's link estimator. The development of 4C is motivated by the fact that PRR based link estimation metrics, such as ETX, tend to capture only long term link quality variations. Similarly, the assumption that PRR based metrics can capture stable links in the network is invalid due to notably frequent variations in wireless link qualities.

4C borrows a number of design concepts from BLE: (1) It works in parallel with an existing link estimator, (2) it is data-driven and uses packet overhearing to predict the quality of links with neighboring nodes, (3) it is receiver-initiated and computes the short temporal quality of a link, (4) it temporarily switches a node's parent to utilize long range bursty links in the network, and (5) regresses back to old parent if the number of consecutively lost packets exceeds a threshold.

The main difference between 4C and BRE lies in how they predict the short temporal quality of a link: The former captures bursts of reliable transmission periods using models trained with specific data traffic patterns instead of a heuristic based approach[3] employed by the latter. Moreover, 4C announces temporary changes in the *parent* across the routing tree after a certain number of successful transmissions, while BRE avoids this to ensure a stable routing topology (cf. Section 4.4.1). The use of highly trained models secures better performance for 4C versus BRE. However, this results in a slightly higher processing cost (i.e. 0.5 ms of network processing delay) for 4C.

4.2.2 ExOR: Opportunistic Routing

ExOR [BM05b] reports a throughput increase of 35% by utilizing long range wireless links in IEEE 802.11 based wireless networks. It uses the broadcast primitive and an agreement protocol among the intermediate nodes that receive a batch of packets for prioritizing the intermediate node closest to the destination for forwarding packets. However, it has a relatively high overhead with regard to computational cost, storage, and communication, which is not feasible in resource constrained sensornets. Opportunistic routing operates on a batch of packets and tries hard to reach a delivery threshold of 90% before falling back to traditional routing for delivering the remaining 10% packets. We share the same spirit as opportunistic routing but differ significantly in detail: Our primary goal is to reduce the number of transmissions in the network. We apply unicast forwarding and hence the next forwarder of the packet is predetermined. Similarly, our approach rapidly falls back to traditional routing to avoid overshooting links with high loss rates. Our aim is to utilize long-range bursty links to increase routing progress and throughput without introducing significant overhead in terms of computation, storage, and communication.

[2]BRE was published in ACM SenSys 2009 and 4C was published in ACM/IEEE IPSN 2011.

[3]The BLE approach was developed after BRE and now we also use a quantitative link estimate instead of a qualitative prediction as originally used by BRE in [ALL+09, ALBL+09].

Property	ExOR	BCP	4C	BRE
New headers	✓	✓	-	×
Backup routing	✓	×	✓	✓
Transparent	✓	–	✓	✓
No. of Tx.	×	×	✓	✓
Throughput	✓	✓	×	✓
Broadcast	✓	×	×	×
Overhearing	✓	✓	✓	✓

Table 4.1 Operational differences between ExOR, BCP, 4C, and BRE: *Transparency* means if the proposed mechanism is independent of the underlying routing protocol and can be integrated with any other routing protocol. *No. of Tx.* and *Throughput* are routing evaluation metrics. *Broadcast* only refers to the transmission mode of the data packets.

4.2.3 Backpressure Collection Protocol (BCP)

BCP [MSKG10] is the first ever implementation of *dynamic back pressure routing* [TE92, pLN10, Nee08, Nee] in which forwarding decisions are made on a per packet basis at each hop. It neither employs any explicit path computation nor an explicit reference to the destination. This allows for greater responsiveness to link variation, queue hot-spots, and node mobility. BCP enhances throughput of collection in sensornets by 60% when compared to the widely used CTP protocol. Similarly, due to its high responsiveness to link variations, it also reduces the average packet transmissions by more than 30%. The main roadblock to BCP's superior performance is this: By generating queue backlog gradients that decrease towards the sink and encode certain utility and penalty information, nodes can make better packet routing and forwarding decisions without the notion of end-to-end routes.

In contrast to BCP, BRE targets the average number of transmissions in the network as its key evaluation metric. It manipulates link burstiness and dynamics to minimize transmissions in the network. Whereas, BCP disseminates queue backlogs to maximize the collection throughput. BRE is a modular extension rather than a full fledged protocol in itself. It protects the simplicity and energy efficiency of protocol design in severely constrained sensornets. The simple design and clarity of BRE's algorithm facilitates its transparent integration with any underlying routing platform, including BCP.

Table 4.2.2 lists the main differences between ExOR, BCP, 4C and BRE.

4.3 System Overview

In this section we present the basic concept of BRE using a simple use case: The collection tree protocols in sensornets. We also discuss the main goals and challenges of integrating bursty links into the routing process.

4.3.1 Basic Concept

Typically, routing protocols in sensornets aim to establish a routing tree: Some number of nodes in the network would advertise themselves as base stations, i.e., as tree roots. All other nodes join the tree with ETX as the routing metric. Figure 4.1 shows an example of such a routing tree rooted at the base station D. A path from source S to the destination D consists of a sub-sequence of immediate parents of each node, for example $S \rightarrow 1 \rightarrow 2 \rightarrow 3 \rightarrow D$. The minimum number of transmissions required by a packet to travel from the source to the destination is four. Now consider a situation in which an intermediate link $S \rightarrow 2$ or $1 \rightarrow D$ has become temporarily reliable. Routing over these links could result in a path sequence $S \rightarrow 2 \rightarrow 3 \rightarrow D$ or $S \rightarrow 1 \rightarrow D$, respectively. Hence, using these links for routing could reduce the total number of transmissions to three in the former and two in the latter case. However, a traditional routing protocol does not make use of such an opportunity because it only uses a long-term link estimate. Hence, this design is inherently unable to realize short-term changes in the link quality. Similarly, even if these short-term changes are captured, traditional routing schemes adapt slowly to ensure routing stability.

In contrast, with a link estimation mechanism in place (e.g. STLE and BLE) to estimate rapidly changing links in the short-term, our proposed technique takes advantage of the availability of such links. In this particular case for example, node 2 overhears the packets addressed to node 1 by source S. After node 2 successfully overhears a certain number of consecutive packets from source S, it informs S about the short-term availability of this link. Thereafter, S examines the burstiness of this link (e.g. by using BLE) and starts forwarding its packets to node 2 thereby reducing the number of overall transmissions for a packet to reach its ultimate destination.

The packet overhearing technique employed in short-term link estimation benefits from the fact that sensornets typically reveal bursty traffic patterns. Common applications [WTV+07, WWAL+05, HKS+04, LW07, DAG03] operate as monitoring environment to detect and often track events. Typically, their occurrence results in long bursts of packets. Hence, they represent a major fraction of the overall network traffic although they occur rarely. In such situations, short-term link estimators, after overhearing the first few packets over a bursty link, identify it as short-term available for transmission.

4.3.2 Design Goals

Our major design goal is to reduce the number of transmissions in the network and increase routing throughput by utilizing long-range bursty links for packet forwarding. However, we seek to achieve our goal without affecting the reliability and stability of traditional routing. Therefore, our approach of transmitting over links with high loss rates faces four key challenges that influence our design decisions.

- Routing over temporarily available links increases the risk of packet loss. Hence, BRE should ensure efficient utilization of the reliable transmissions periods in bursty links. Similarly, it should provide a backup mechanism when there is no bursty link available for transmission. Otherwise, packet forwarding

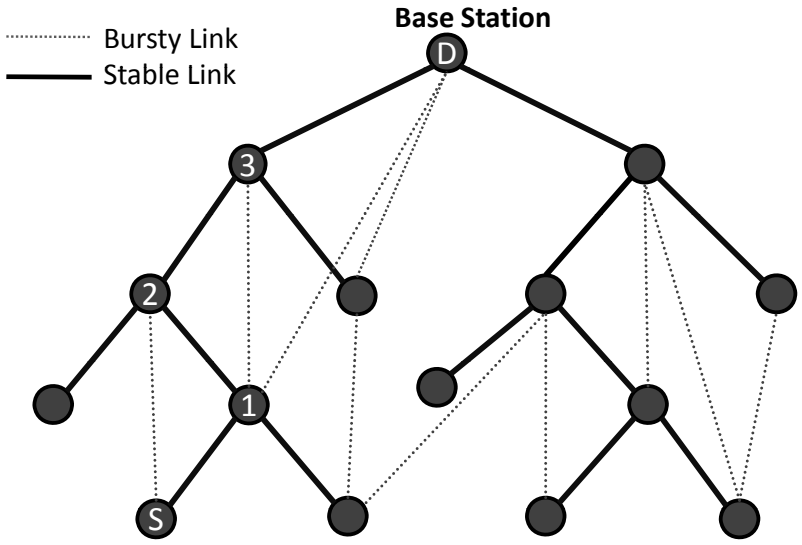


Figure 4.1 Bursty links provide routing shortcuts that can significantly reduce the hop count and the number of transmissions from source to destination.

over such links might increase the number of overall (re)transmissions in the network.

- Only the bursty links that offer better routing progress and do not disrupt the stability of the underlying routing topology shall be identified for routing. Failure to meet this requirement could result in typical routing problems such as loops and network partitioning.

- BRE should be lightweight and resource sensitive in terms of computation, storage, and communication. For example, introducing additional routing tables and computational complexity is prohibitive because it would consume a significant amount of storage and computational energy.

- BRE should seamlessly integrate with existing routing protocols and link estimators. Its use shall not affect applications and services running on top of routing protocols. However, applications must be amenable to path changes induced by BRE and should not expect a static routing path to the destination.

4.4 Bursty Routing Extensions

In this section we discuss the design of bursty routing extensions (BRE) in detail. We first present the algorithm that enables packet forwarding over long-range bursty links. We then show the integration of short-term link estimators and BRE with existing routing protocols and link estimators. Finally, we discuss the major design challenges and how we address them in our implementation.

4.4.1 Algorithm

We define three roles for nodes in the network:

Algorithm 1: BRE Algorithm

```
// Overhearing Node . . .
input  : Neighbor ID and neighbor's parent ID
output: The decision whether or not announce this link as available.
if pathETX(neighborsParent) < myPathETX) then
  | sendAnnouncement(neighbor);
else
  | ignore(neighbor)
//
// Sender Node . . .
input  : An announcement from overhearing node
output: Decision whether or not change parent.
if MAC3(neighbor) > THRESHOLD then
  | routingmode ⟵ BRE;
  | myparent ⟵ neighbor;
else
  | ignore(announcement)
```

- *sender-node*: the node which is actively sending or forwarding packets
- *parent*: the parent of any sender-node in traditional routing and
- *overhearing-node*: node(s) which can overhear the communication between the sender-node and its parent. A node in the network can assume any or all of these three roles at a time.

Algorithm 1 shows the pseudocode for BRE. It has the following four phases:

4.4.1.1 Link Discovery

When an overhearing-node declares a link as reliable for transmission — successfully overhearing three consecutive packets sent by a sender-node to its parent (cf. Section 3.4.2) — it queries its routing table for the path-ETX of the packet's destination, i.e., the parent of the sender-node. If the path-ETX of the parent-node is greater than that of the overhearing-node, the overhearing-node declares this *unused* link between itself and the sender-node as active. Consequently, the active bursty link can offer a better routing progress than the traditional path used by the sender-node. However, if the path-ETX of the parent-node is not known or less than the path-ETX of the overhearing-node, the overhearing-node temporarily ignores the sender-node. In our example in Figure 4.1, it would be node 2 overhearing the communication between node S and its parent 1.

4.4.1.2 Link Announcement

If the path-ETX of the parent-node is greater than that of the overhearing-node, the overhearing-node informs the sender-node about the active bursty link (cf. Figure

4.1, node 2 informs node S about the active bursty link between them). It volunteers to become the temporary parent of the sender-node as long as this bursty links remains active. The path-ETX information used by BRE at the overhearing-node can easily be obtained by using the neighborhood information maintained by any traditional routing protocol.

We assume that there is a high probability that the original parent of the sender-node is also a neighbor of the overhearing-node. This is because the overhearing-node can listen to the ongoing communication between the sender-node and its parent. An alternative approach to remove this neighbor-table dependency is to include the path-ETX of the parent in each packet. However, this approach introduces 1 byte overhead in each data packet.

Additionally, the link announcement message, sent by the overhearing-node to the sender-node, establishes a simple check to test for link-asymmetry.

4.4.1.3 Routing Mode

The sender-node, after receiving the announcement from the overhearing node, queries the link estimator about the burstiness of this link. Consequently, if the burstiness of a link exceeds a certain threshold, the sender-node makes the overhearing-node its temporary parent and starts forwarding packets to it (cf. Figure 4.1, S forwards its packets to 2). However, this information is not propagated by the routing protocol to its descendant nodes because these short term changes would trigger further parent changes down the tree. Eventually, it might destabilize the routing protocol and introduce loops. This is one of the primary reasons why stability prevails over adaptability in today's routing protocols and link estimators. Hence, our routing strategy supplements their design considerations.

The main disadvantage of this approach is that BRE operates greedily. Although this approach is still effective for enhancing routing progress when compared to traditional routing, it does not guarantee the use of the optimal path currently available in the network. For example, a node may change its parent based on the recommendation of BRE. However, it is possible that along the traditional path the sender-node remains unaware of the availability of an even better bursty link currently reliable for transmission. Nonetheless, we believe that our approach strikes an efficient trade-off between routing stability and performance adaptability.

4.4.1.4 Link Unavailability

The sender-node declares a link unavailable for transmission after it fails to receive a number of acknowledgments (see Section 3.4.2) for the data packets sent over the bursty link. The sender-node will then regress to traditional routing until it receives another link announcement.

4.4.2 Integration with Routing Protocols

Our goal is to enhance routing performance without affecting the stability and reliability of traditional routing protocols. Therefore, we neither replace the existing link

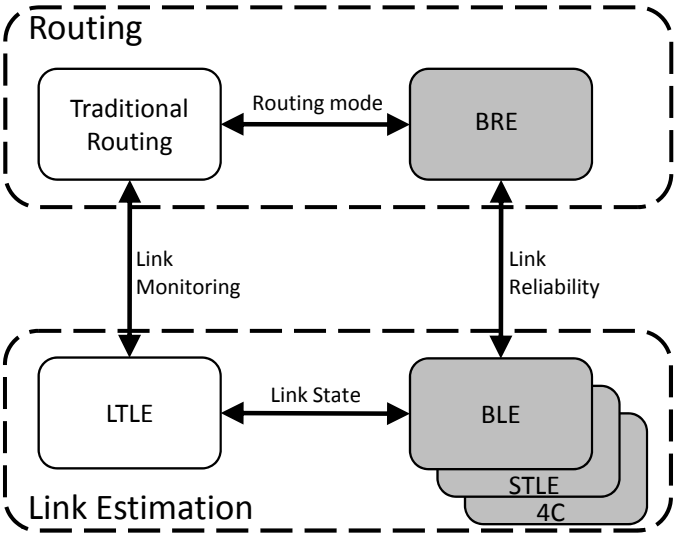


Figure 4.2 Design of BRE: It owns a modular design and can host different types of link estimators and routing approaches.

estimators nor alter the stable routing topology maintained by traditional routing protocols. Rather, our approach adds an additional component to the system architecture that assists routing protocols and link estimators in identifying the previously ignored class of bursty links which can enhance routing performance. BRE seamlessly integrates with existing routing protocols and link estimators because (i) it does not introduce new routing tables, and (ii) it does not require any modifications to the packet headers.

We define two routing modes, a *bursty mode* and a *traditional mode*. In *bursty mode* packets are forwarded over the active bursty links identified by a short-term link estimator. Conversely, in *traditional mode* packets are forwarded along the path chosen by the regular routing algorithm.

The integration of short-term link estimators and BRE into traditional routing protocols only requires three interfaces.

- The first interface is between the link estimator and BRE. Using this interface, the link estimator informs BRE about the availability of a potentially beneficial bursty link.
- The second interface is between BRE and routing protocols. This interface is used to switch between different routing modes and to access the neighbor table maintained by the routing protocols to inquire the path-ETX of neighboring nodes.
- The third interface is between the two link estimators to share link state information. The main purpose of this interface is to ensure that the short-term link estimator does not estimate a high quality link already used by the routing protocols.

The design of BRE is not specific to BLE. It can accommodate different types of link estimators that aim at utilizing routing shortcuts offered by intermediate links. For

example, Liu and Cerpa [LC11] report an improvement in average cost of delivering a packet by 20% to 30% when integrating 4C (cf. Section 4.2.1) with BRE. Similarly, we also integrated STLE [ALL+09] with BRE to minimize routing delivery costs. Figure 4.2 shows the major design components of BRE and their integration with traditional routing protocols.

4.4.3 Design Challenges

Reliable end-to-end packet transmission and stable network topology are the basic requirements of wireless routing protocols. One of the major concerns that surfaces with routing over intermediate links is its impact on the routing stability and reliability. Therefore, any approach that attempts to route packets over intermediate quality links needs to alleviate these concerns. Our goal is to benefit from the increased routing progress of specific bursty links. However, we do not want to deteriorate the stability and reliability of wireless routing. In the following sections we address the challenges that stem from packet forwarding over bursty links.

4.4.3.1 Reliability

To ensure high end-to-end reliability, we have three built-in mechanisms in our approach. First, we eliminate all the bad links that rarely transmit a packet by keeping a recent history of transmission characteristics and waiting for three successful transmissions before declaring a bursty link active. Secondly, our approach employs an aggressive back-off technique to stop transmitting over a bursty link even after a single packet loss. Both these mechanisms ensure that we do not overshoot a bursty link. Therefore, unlike reactive routing protocols [PRD99, JM96, GE00] in which route discovery is typically triggered by a route break and route timeouts [RSBA07b], our approach promptly reacts to the changes in link quality.

Finally, as a backup, we use traditional routing and its retransmission mechanisms to deliver the packets that failed over bursty links. The analysis of our experiments in Section 4.5.3.2 rationalizes that our approach indeed does not affect the reliability of traditional routing.

4.4.3.2 Stability and Adaptability

Routing stability prevails over performance adaptability in traditional routing protocols [RSBA07b]. Typically, route evaluation depends on the rate at which beacons are exchanged in traditional routing protocols. However, data is typically exchanged at much higher rates than beacons. Therefore, traditional routing protocols fail to recognize the route quality fluctuations that occur at shorter time scales proportionate to the data exchange rates.

In contrast, always picking the optimal path can itself be detrimental for network performance due to the following reasons: (1) the resulting instability can lead to routing problems such as loops and (2) the overhead associated with active link estimation at shorter time scales is not acceptable for resource constrained sensornets in terms of energy and bandwidth.

BRE finds a suitable trade-off between stability and performance adaptability. Its route evaluation is dependent on the time and the rate at which the data is transmitted and independent of the rate at which beacons are exchanged in the network. Hence, a short-term link estimator monitors a link by overhearing data packets that, due to the broadcast nature of wireless medium, are received in any case. As a result, it ensures link estimation at a high resolution in time with only a small communication overhead. Moreover, local optimizations performed by our adaptive routing strategy in response to the short-term link quality variations are not distributed among other nodes in the network. Therefore, our approach preserves the routing stability by sustaining the routing topology laid down by traditional routing protocols. Hence, as discussed in the following section, our approach does not introduce routing problems such as loops.

4.4.3.3 Loops

Loops (or cycles) are a common routing problem in wireless networks which occur due to sudden changes in the routing topology. Loops occur when, due to sudden loss of connectivity to the current parent, a node selects a significantly higher ETX route that also contains a descendant node. A loop is detected when a receiver node finds that its ETX is higher than the ETX of the sender of the packet.

Our adaptive routing approach inherently prevents routing loops. The temporary parent selection mechanism ensures that an overhearing-node is only selected as a new temporary parent if it has a lower path-ETX than the current parent. Additionally, our approach operates locally and does not inform the descendant nodes about the temporary changes made in the parent selection. Hence, our routing strategy, although highly adaptive, does not amplify the looping problem because it preserves the underlying routing topology. Apart from the rare occurrences of loops in traditional routing, we have not observed any additional loops during the evaluation of our approach. Therefore, the loop detection mechanism employed by traditional routing protocols is sufficient, as the integration of BRE does not escalate the occurrence of loops in traditional routing.

4.4.4 Duplicate Transmissions

Duplicate transmissions occur due to the loss of link level acknowledgements in asymmetric links. Routing protocols, such as CTP, therefore use bidirectional link quality estimates to find the best paths to the destination. Currently, we employ a very simple strategy to test the asymmetry of a link: The announcement sent by the overhearing-node to inform the sender-node about the short-term availability of a bursty link (see Section 4.4.1) serves as simple test to identify link asymmetry. The performance analysis in Section 4.5.3.1 shows the viability of this simple approach. Moreover, BRE utilizes the duplicate suppression mechanism employed by traditional routing protocols to squelch duplicate packets from overwhelming the network.

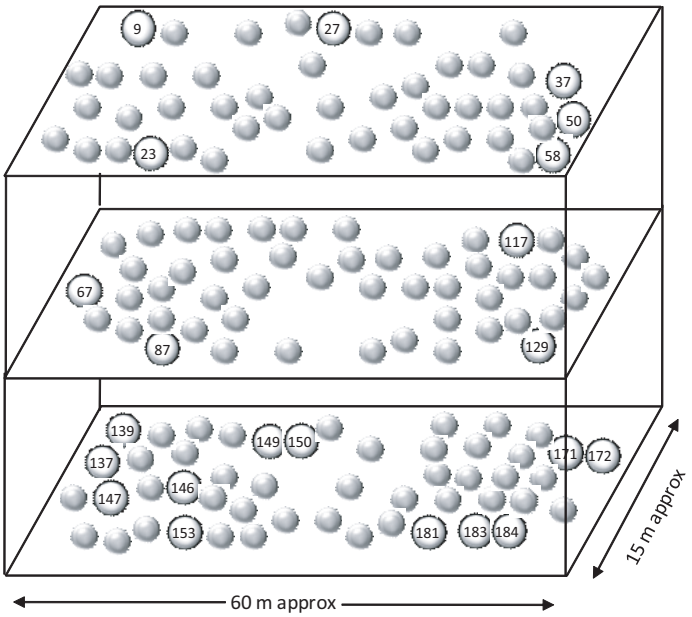


Figure 4.3 An abstract representation of the MoteLab topology on three different floors. The figure does not show the walls between rooms. The node IDs are only shown for the nodes that were used either as senders or collection roots during our experiments

4.5 Evaluation

In this section we evaluate the performance of BRE when compared to CTP [GFJ+09], the most widely used collection protocol for sensornets shipped with TinyOS. We divide our evaluation in three main sections. After describing the implementation details and experimental setup, we evaluate the impact of BRE on the routing cost and throughput. We give a detailed account of the timing properties of the bursty links used by BRE to enhance routing performance. Next, we evaluate the impact of recent transmission history of a link on the performance of BRE in a multihop setting. This evaluation further validates the threshold of three-packets in calculating the CPDF in our prototypical implementation. Next, we conclude our discussion by giving a detailed account of the overhead introduced by BRE.

The data analysis mainly focuses on routing issues such as transmission costs, delivery reliability, and throughput. Experimental studies, such as [SKAL08] and [BLKW08], give further insight into the properties of intermediate and bursty links.

4.5.1 Implementation

We have implemented BRE in nesC [GLvB+03a] for TinyOS 2.x. The prototype implementation of BRE is integrated with CTP. CTP uses the Four Bit Link Estimator (4BLE) as its link estimation component. Although CTP is explicitly designed for relatively low data rates, we observed that it is capable of handling high traffic rates as well (i.e. it can deliver a packet every 25 to 30 ms in a multihop network). Moreover, CTP has a very robust retransmission mechanism that ensures high delivery reliability. This property of CTP allows us to thoroughly evaluate the impact of BRE on the reliability of traditional routing protocols. However, BRE is not bound

Experimental Class	Intermediate Links %	Forwarders %	Candidates %	Node Density	Potential Neighbors	Candidate Neighbors
Horizontal	33.3	94.8	90.2	15.0	11.2	8.6
Vertical and Diagonal	36.5	93.4	88.4	23.2	14.8	8.5
Nearby	14.2	86.2	79.3	16.3	9.6	4.0

Table 4.2 MoteLab statistics for experimental parameters defined in Table 4.5.1. The statistics for *Intermediate Links, Node density, Potential Neighbors,* and *Candidate Neighbors* were collected by randomly selecting 10 motes from different locations (i.e. corner, center) in the test-bed. The statistics for *Forwarders* and *Candidates* were collected by running BRE on all the motes (sending a packet every 5 seconds) with a collection root (i.e. mote 183), located at one corner of the network.

to any specific routing protocol. It can easily be integrated with BVR [FRZ+05] or other routing strategies that support higher data rates for bandwidth limited systems. Such strategies could, for example, merge multiple data frames into a single link layer packet.

4.5.2 Experimental Setup

The majority of our experiments were executed on MoteLab, a widely used sensor testbed at Harvard University. MoteLab is an indoor deployment of 190 TMoteSky [PSC05] sensor motes on three different floors (see Section 5.4.1 for further details). However, due to the difficulty of maintaining such a large test-bed, only 142 motes were available to us at maximum. All our experiments had the following common characteristics unless stated otherwise: (1) Motes transmit at full transmission power, i.e., O dBm. (2) We use an inter-packet interval of 250 ms (results are presented for different inter-packet intervals as well). (3) We use the default $\alpha = 9$ for WMEWMA based estimation [WTC03] in 4BLE, and IEEE 802.15.4 channel 26. (4) Each experimental run lasted for 30 minutes.

To ensure the validity of our MoteLab results, we re-ran our experiments on TWIST [HKWW06], a 100 node TMoteSky-based testbed at TU Berlin. TWIST is a high-density grid-like deployment with an inter-mote spacing of 3 meters (see Section 5.4.1 for further details). Therefore, to create a reasonably large multihop network, we reduced the transmission power to -15 dbm for our experiments on TWIST. The other characteristics are identical to our experiments on MoteLab.

4.5.3 Performance

In this section we thoroughly evaluate the performance of BRE in terms of transmission cost, throughput, and reliability. Our major performance benchmark is to reduce the number of transmissions in the network by enhancing routing progress. Figure 4.3 shows a schema of the MoteLab topology[4] and highlights the motes that

[4]This is an abstract representation of the MoteLab topology. A detailed topology and connectivity graphs can be found at http://www.MoteLab.eecs.harvard.edu

Name	Description
Horizontal	Source and destination at the opposite ends on the same floor. Only the motes on the same floor were used for this class of experiments (e.g node-pair $9 \rightarrow 50$).
Diagonal	Source and destination on different floors and on the opposite ends. All the motes in MoteLab were used (e.g. $137 \rightarrow 50$).
Vertical	Source and destination on different floors but on the same end. All the motes in MoteLab were used (e.g. $183 \rightarrow 50$).
Nearby	Source and destination are nearby to each other but surrounded by a high density of nodes. Only 30 to 50 neighboring motes were used (e.g.$153 \rightarrow 183$).
Intermediate Links	The percentage of links in the network with average PRR less than 90%
Forwarders	The percentage of the overhearing-nodes in the network that can overhear a data packet and have a lower path-ETX than the path-ETX of the parent of the sender
Candidates	The percentage of the overhearing-nodes in the network that can overhear three consecutive data packets and have a lower path-ETX than the path-ETX of the parent of the sender
Node Density	Number of neighbors that can overhear a node's data packet
Potential Neighbors	Number of neighbors that can overhear a node's data packet and have a lower path-ETX than the path-ETX of its parent.
Candidate Neighbors	Number of neighbors that can overhear three consecutive data packets from a node and have a lower path-ETX than the path-ETX of its parent.

Table 4.3 Description of experimental classes and parameters presented in Table 4.5.

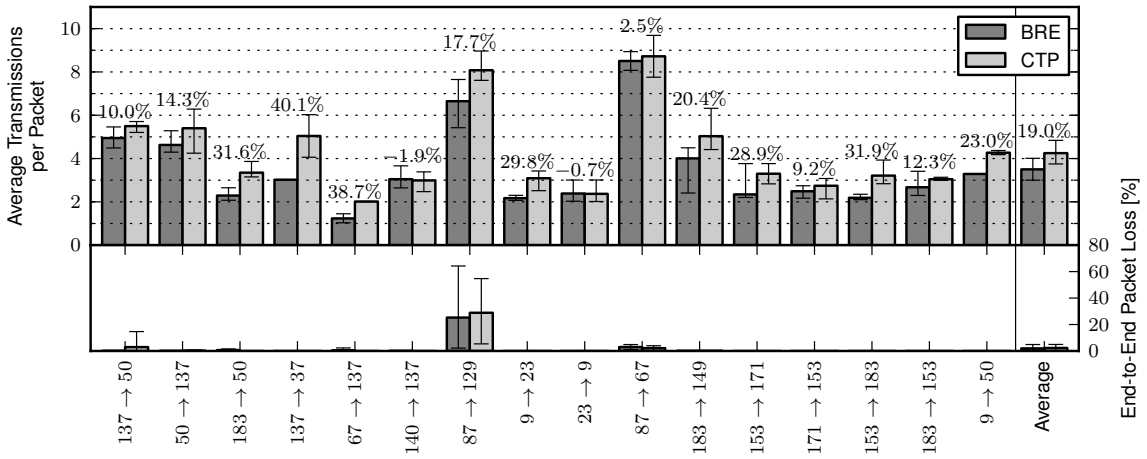


Figure 4.4 Transmission cost reduction and reliability comparison of BRE and CTP. The graph above shows average number of transmissions per packet using BRE and traditional CTP for our experiments on MoteLab. The graph below shows end-to-end packet loss for the same experiments. The bar represents a node pair's average of five experiments. The error bars represent the highest and the lowest average of the five experiments. The inter-packet interval is 250 ms. For these experiment, the average retransmissions is 8.05% for BRE and 3.5% for CTP. The reduction in the number of transmissions in the case of BRE is mostly due to the reduction in the number of hops.

were used as senders and receivers in all our experiments. We define four different experimental classes - namely horizontal, vertical, diagonal and nearby - to comprehend different network sizes and topological and physical scenarios (see Table 4.5.1). Our mote selection as a source and destination is also based on the these experimental classes.

Before presenting our performance evaluation results, we demonstrate important topology characteristics that describe our analysis and allow for a deep understanding of the results that follow. These parameters are presented in Table 4.5 and their descriptions are presented in Table 4.5.1.

The high percentage of *Forwarders* and *Candidates* in Table 4.5 shows that a large number of nodes in the network can be utilized in our bursty forwarding approach. Table 4.5 testifies to the fact that more than 60% (i.e. 11.2 potential neighbors out of 15 neighbors in class *horizontal*) of a node's immediate neighbors had a better path-ETX than the original parent. Correspondingly, out of these *potential neighbors*, more than 70% (i.e. 8.6 candidate neighbors out of 11.2 potential neighbors in class *horizontal*) could even overhear three consecutive data packets. It means that these neighbors were not selected as a parent only because of a poor long-term quality estimate of their links with the sender. Algorithms that assess links based on average PRR, like most current approaches, do not use such a link, not even while it is in its good state. The high average of the measured packet loss rate based on broadcast beacons prevents the recognition of good transmission periods in such links.

Another observation is that, with the decrease in the number of intermediate links in the network, the number of *potential neighbors* and *candidate neighbors* also decreases (see Table 4.5 for class *nearby*). Although the *node density* of experimental

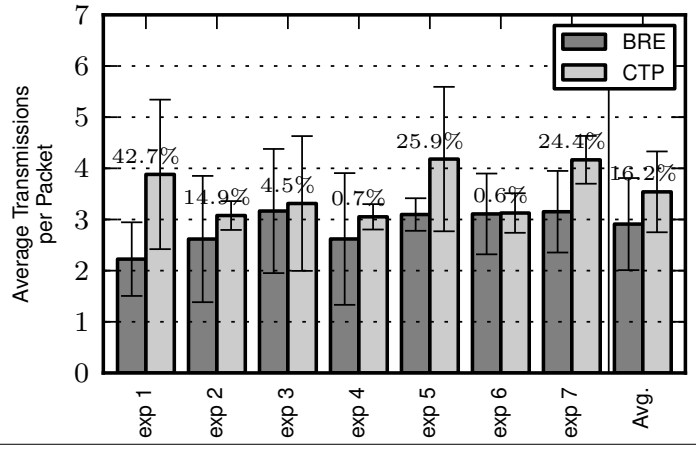


Figure 4.5 Average number of transmissions per packet for single experimental runs on TWIST. The error bars in this case represent the standard deviation. The results are similar to the MoteLab experiments.

class *nearby* is higher than class *horizontal*, the class *nearby* has a smaller number of *candidate neighbors*. It means that CTP (based on the link estimates of 4BLE estimator) indeed selected the best neighbor as a parent from the neighbors with high quality links. This information supports the hypothesis in [WTC03] that WMEWMA-based link estimators performs well when estimating good links. However, they perform poorly when estimating intermediate links.

4.5.3.1 Transmission Cost

We compare the transmission cost of BRE with CTP. Figure 4.4 shows our results for 16 randomly selected node-pairs as senders and collection roots. To observe the stability of results over time, we repeated our experiments for BRE and CTP three to five times for each of the 16 node-pairs depending upon the difference in number of transmissions. For example, if the average number of transmissions in the case of CTP for a particular node-pair differed by more than one transmission after three experiments, we executed the experiments five times. By reprogramming all the motes involved in an experiment for each experimental run, we enforce CTP to re-establish its routing tree. As a result, we intensively validate our results for a particular node-pair. In most of the cases BRE performs better than CTP, averaging to approximately 19% overall reduction in the transmission costs, i.e., the total number of transmissions from source to destination for single node-pairs.

Although BRE decreases the total number of transmissions in the network by reducing the number of hops, it increases the number of retransmissions when compared to CTP. This is because it risks transmission over links with high loss rates and retransmits all the lost packets via traditional routing. The percentage of retransmissions is 8.05% for BRE and 3.5% for CTP in the experimental results presented in Figure 4.4.

To see if these results carried over to other networks, we repeated our experiments on TWIST[5] using a lower transmission power of $-15dBm$ (see Figure 4.5). These

[5]The privacy rules of TWIST did not allow us to show the exact locations and IDs of the node pairs used in our experiments. We used the motes placed on the opposite corners (e.g. south-east

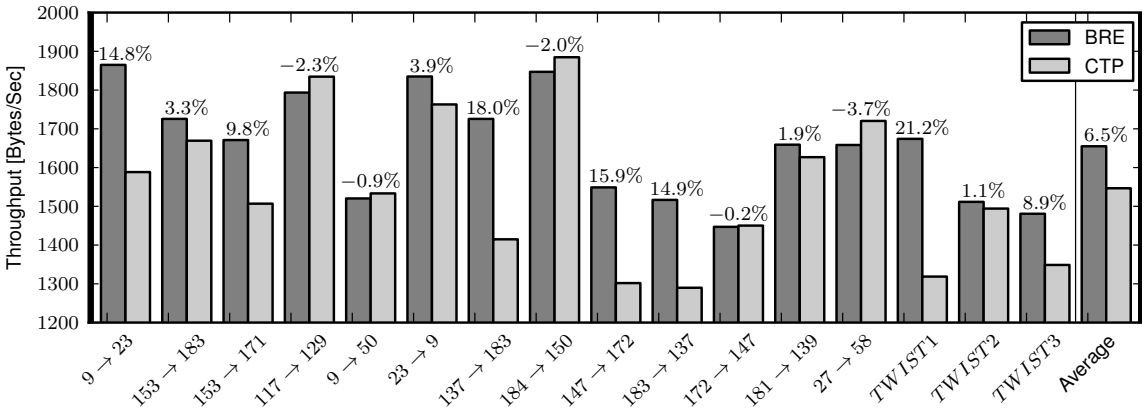


Figure 4.6 Evaluation results for measured throughput on MoteLab and TWIST. BRE increases routing throughput of traditional routing in most of the cases. The last three bar-pairs show the results for our experiments on TWIST.

results are similar and sometimes even better than the results for our experiments on MoteLab. The presented results for an overall of 23 different node-pairs from two different testbeds demonstrate the feasibility of our approach.

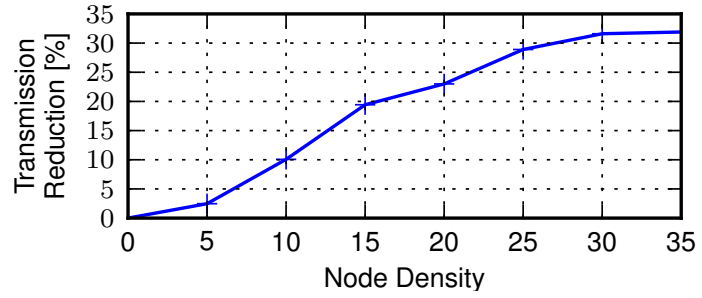
There are only a few cases (e.g. node pair 140 → 137) in which CTP is marginally better than BRE. This is due to a simple design trade-off in our prototype implementation of BRE: For analyzing the precise impact of transmission over intermediate links, currently, we always select an intermediate link without assessing the risk of transmission over such a link. For example, always selecting an intermediate link which has a higher loss rate and only offers a mere 0.1% reduction in transmissions is not always feasible. Frequent failures of transmission over such a link can increase the overall number of transmissions in the network, as depicted in Figure 4.4 for node-pairs 140 → 137 and 23 → 9.

4.5.3.2 Reliability

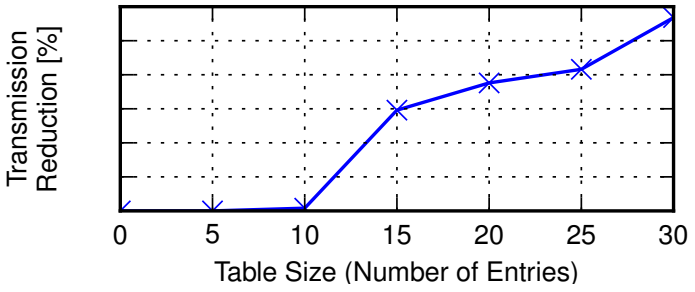
Figure 4.4 also presents the end-to-end packet loss for our experiments. In most cases, packet loss is negligible. From these results, it is fair to conclude that BRE does not affect the reliability of the underlying routing protocol and at the same time reduces the number of transmissions in the network. Using the adaptive routing strategy, BRE makes an attempt to forward packets over long-range bursty links. However, when it fails to transmit a packet over a bursty link, it backs off and allows CTP to retransmit the packet over the traditional path.

The only measurable end-to-end packet loss observed in our experiments is for the node-pair 87 → 129 and 87 → 67. We regard these two node-pairs as a sanity check for BRE, as they possess a very lossy path. The average number of hops traversed by each packet for these node pairs is 3 and 4 respectively. However, the average number of transmissions per packet is approximately 8. Therefore, the average link quality is less than 50% in both cases. The upper graph in Figure 4.4 shows that BRE performs better than CTP even in such lossy scenarios. Similarly, the average

and north-west corners) and different floors as senders and collection roots in the grid-like TWIST deployment.



(a) Impact of node density on the performance of BRE for MoteLab.



(b) Impact of Table-size on the performance of BRE for MoteLab.

Figure 4.7 Factors limiting the performance of BRE. Higher node density increases the probability of finding a routing shortcuts offered by neighboring nodes with busty links. Larger routing table increases the probability of finding the original recipient of the packet for path-ETX comparisons.

end-to-end packet loss for BRE in the case of 87 → 129 is less than in traditional routing. However, as discussed in Section 4.5.4.2, these two node pairs incur a higher transmission overhead.

Although these node-pairs are surrounded by a large number of motes, as shown in Figure 4.3 as well as in official MoteLab connectivity maps, we calculated the node density and link qualities for mote 129 to find the exact reasons of this high packet loss. The average PRR for node 129 was less than 40% for all the neighbors, and node density was 4.

4.5.3.3 Throughput

The two key factors that impact routing throughput in a multihop sensornet is the number of retransmissions and the number of hops. Routing throughput can be increased by minimizing the number of retransmissions for a packet to travel from source to destination. Similarly, each hop traversed by a packet also negatively impacts the throughput. The modest computational capability of a mote and protocol-specific considerations result in additional delays, such as packet processing requirements and CSMA-backoff waiting time. BRE adapts to both these key factors. Although it slightly increases the number of retransmissions in the network, the significant reduction in the number of hops contributes to increasing routing throughput (see Figure 4.6).

Protocol	Transmissions per packet	Throughput bytes/sec
BRE	4.34	1677
CTP	5.25	1583
Strawman	6.88	1793

Table 4.4 Summary of the results for BRE and CTP when compared to a strawman. Strawman increases the throughput and the number of transmissions by a factor of 1.06 and 1.8 respectively, when compared to BRE.

CTP is not an ideal candidate for throughput measurements as it is a reliable routing protocol originally developed for relatively low traffic rates [RGJ+06]. We still believe that it can provide us with useful hints about the significance of our approach in terms of routing throughput. Furthermore, we use CTP because we wanted to evaluate the maximum throughput without affecting the delivery reliability - a key property of sensornet routing. Our technique for evaluating throughput is to send a packet by calling the *Send* interface of CTP immediately after CTP signals a *SendDone* event for the previous packet. Figure 4.6 presents our throughput evaluation results[6] for MoteLab and TWIST. It shows that in most of the cases, due to the reduction in the number of hops, BRE improves the routing throughput - with a maximum improvement of 21%. We expect our approach to be more beneficial if integrated with routing protocols supporting high traffic rates. Moreover, the room for throughput improvement in a bandwidth limited system, like a sensornet, is very limited: Langendoen [Lan06] reports a maximum link throughput of 3KB/s for CC2420 without routing in TinyOS. Therefore, in addition to our primary goal of reducing the number of transmission, the throughput increase revealed in Figure 4.6 is a welcome improvement in a multihop sensornet.

Concluding our performance evaluation results, BRE reduces the number of data transmission in the network without affecting delivery reliability. Additionally, by reducing the number of hops for a packet to reach from source to destination, it also enhances routing throughput.

4.5.3.4 Comparison with Strawman

In this section we compare BRE with a simple strawman approach - where if a node with lower path-ETX overhears a packet, it simply forwards it immediately, without updating any table. The duplicate packets that arrive along the standard path are later dropped by the overhearing nodes. Comparison with a strawman allows to understand the limits and trade-offs between the transmission cost and throughput of BRE (see Table 4.5.3.3). We performed our experiments by selecting a single node (node 183) as a root at one corner of Motelab, while other nodes (numbered in Figure 4.3) sending one at a time a total of 500 packets each. The results clearly show that, while strawman improves the routing throughput by 6%, it increases the number of transmissions by a factor of 1.8 versus BRE.

[6]These results are not comparable for corresponding node pairs in our performance measurement results in Figure 4.4. The reason is that all our experiments were carried out in a span of 3 months. The MoteLab topology changed significantly during that period. This is also the reason that we had to use different node pairs for throughput evaluations.

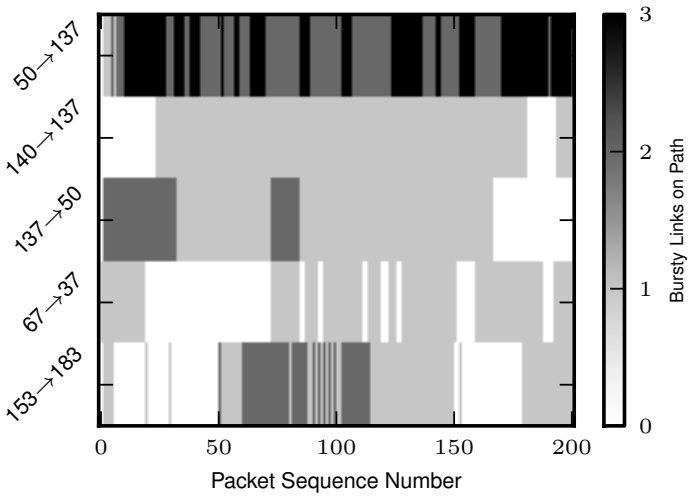


Figure 4.8 Timeliness of bursty links for 50 second empirical traces for selected node-pairs. The graph shows the variability in the duration for which intermediate links are reliable. Most of the successful packets took one or more bursty links on the path from source to destination. Only the white segments in the graph represent complete packet transmissions on traditional path.

4.5.3.5 Node Density and State Maintenance

In this section we analyze how node density and the state of neighboring nodes maintained by BRE impact its performance. Node density positively impacts the performance of BRE as it has more neighboring nodes to choose from. Similarly, higher density increases the probability of finding neighboring nodes with lower path-ETX. This trend is shown in Figure 4.7(a). A similar trend can also be seen when comparing different experimental classes presented in Table 4.5 and the corresponding node-pairs in Figure 4.4. The node-pairs that belong to high-density experimental class *vertical and diagonal*, such as 137 → 37 and 67 → 137, achieve higher reduction in transmissions.

Finally, we evaluate the impact of table size on the performance of BRE. As discussed in Section 4.4.1, the presence of the original destination of the packet in the routing table of overhearing-node is necessary for path-ETX comparisons: The greater the size of the table, the higher the probability of finding the original destination of the packet (i.e. parent of the sender node). Figure 4.7(b) shows that BRE achieves a very small performance gain for neighbor-table sizes of less than 10 entries on Motelab. However, higher table sizes (e.g. 20 entries) benefit BRE by increasing the probability of finding the original destination of the packet in the table. This neighbor table dependency can be removed completely by including path-ETX of the parent in each data packet (cf. Section 4.4.1). However, this approach and other routing strategies are beyond the scope of discussion in this dissertation, and therefore, we regard them as a future work.

4.5.4 Intermediate Link Characteristics

After evaluating the performance of BRE, we now analyze the properties of bursty links in more detail. First, we examine the level of correlation between transmission

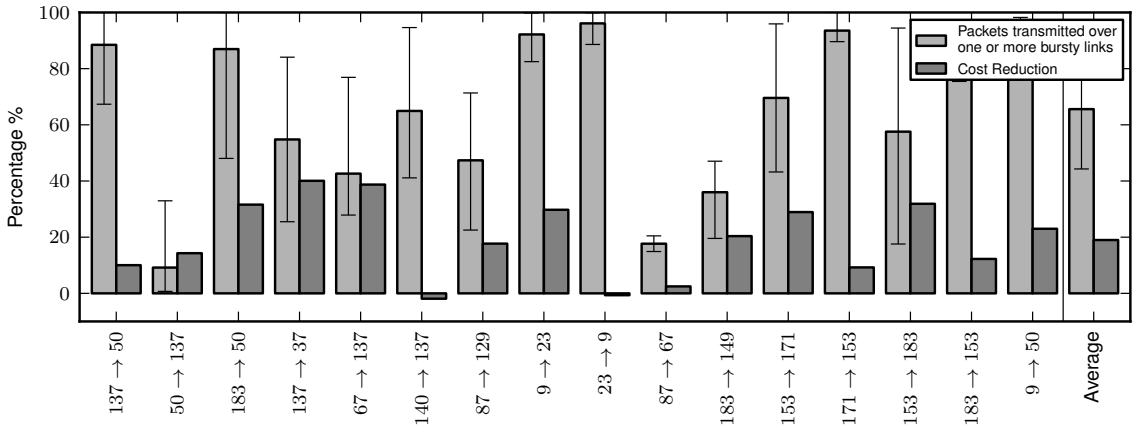


Figure 4.9 Average number of packets transmitted over one or more bursty links vs. reduction in the number of transmissions for the node-pairs as in Figure 4.4. A large number of packets took one or more bursty links on the path from source to destination in most of the experiments. There is no correlation between the number of packet transmissions over bursty links and the reduction in overall transmissions. For example, in 23 → 9, about 100% of the successful packets took one or more bursty links but did not reduce the number of transmissions in the network. However, in 50 → 137, up to 35% packet transmissions over intermediate links result in 15% reduction in the number of transmissions.

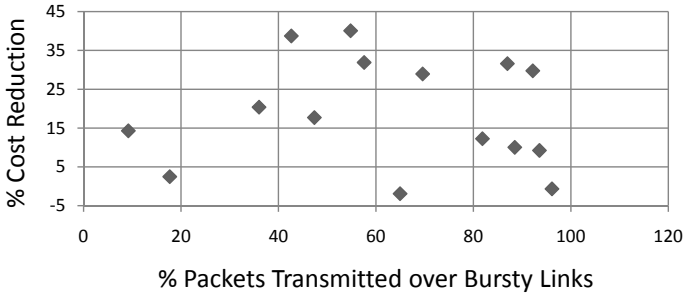


Figure 4.10 Correlation between the number of packet transmissions over bursty links and the reduction in overall transmissions.

reduction and the number of bursty links used for transmission. Next, we present empirical traces from our experiments to investigate the nature and timeliness of intermediate links used for packet forwarding. Finally, we evaluate the impact of different transmission speeds on the performance of BRE.

4.5.4.1 Transmissions Cost vs. Intermediate Links

In this section, we observe the number of overall packets that were transmitted over one or more bursty links throughout the duration of an experiment. Figure 4.9 shows that in most of our experiments, more than 50% of the packets were transmitted successfully over one or more bursty links. From these results, it is adequate to conclude that bursty links exist over time in the network and that they are short-term reliable for transmission.

Figure 4.10 depicts that there is a weak correlation between cost reduction and the number of used bursty links. This fact can also be seen in figure 4.9 by considering

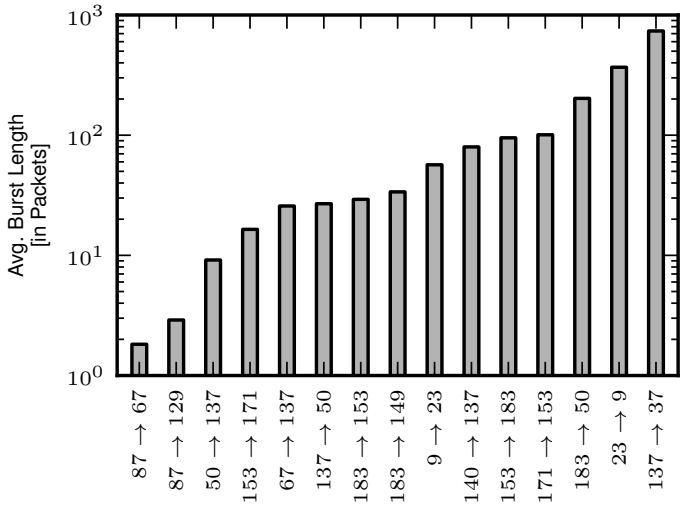


Figure 4.11 Availability of Bursty Links in packet durations. This figure depicts that even relatively long-term (i.e., 750 packet durations) reliable links were not utilized by CTP. It also shows the limited transmission overhead incurred by BRE.

node-pairs 50 → 137 and 23 → 9. The node-pair 50 → 137 achieves 15% reduction in transmission where only 35% packets take one or more bursty links. Whereas, node-pair 23 → 9 achieves no reduction in transmissions even when 100% of the packets were transmitted over one or more bursty links.

4.5.4.2 Timeliness

Another property of bursty links that we investigate is timeliness: how often do they occur and for how long are they active. Figure 4.8 presents empirical traces from our performance evaluation experiments. It shows that bursty links are regularly available over time and are reliable for variable durations. Figure 4.11 shows the average consecutive packet transmissions over bursty links in each of our experiments. Some of these links are active for only a few milliseconds (e.g. 153 → 183), while others for seconds and even minutes (e.g. 140 → 37 in Figure 4.8). However, due to the slow adaptivity of traditional routing, i.e. CTP, even these relatively long-term reliable links with higher routing progress would not be utilized. Figure 4.12 shows the cumulative distribution of the burst lengths for all the experimental results presented in Section 4.5.3.1.

4.5.4.3 Inter-packet Intervals

We investigated the impact of different inter-packet intervals on the performance of BRE. Figure 4.13 shows that the reduction in the number of transmissions decreases with an increase of the inter-packet interval. This is because sending packets at higher rates over bursty links maintains a strong correlation between their success or failure providence. While by sending packets further apart, the packet loss during a certain measurement period becomes independent [SKAL08]. Thus, the observation that the success probability of the next transmission after three consecutive successes is higher (i.e. $CPDF(3) = 0.9$) does not strongly hold for very low transmission

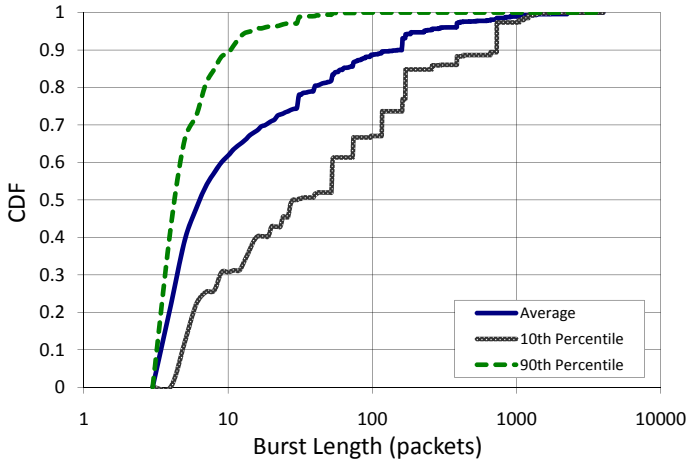


Figure 4.12 Cumulative distribution of packet bursts for all the experimental results presented in Section 4.5.3.1.

rates. It means that at lower transmission speeds, it is less probable that the link estimator declares a bursty link as active in time. Nonetheless, with inter-packet intervals as high as one second, BRE still offers a 5% improvement when compared to traditional routing.

However, as discussed in Section 4.3.1, we target sensornet applications with bursty traffic patterns. Such bursty traffic patterns are typically observed in tracking, monitoring, and surveillance applications. In these applications, the inter-packet interval is expected to be much lower than one second during peak traffic times, i.e., the times when the motes are triggered to track or monitor an activity and report it to the base station.

4.5.5 A Sanity Check for BLE Thresholds

The experimental results presented in Section 3.4.2 conclude that the CPDF(n) increases as the number of preceding successfully received packets over a link increases. Moreover, it concludes that three consecutive successful packet receptions over a link increase the CPDF(3) of the next packet to 0.9, and that a history-size threshold of 3 is a sufficient value to discover active bursty links. We grasp these observations to find (1) if these results holds in different testbed environments, such as MoteLab and TWIST, and (2) to verify the applicability of these results at the routing layer over multiple hops.

Figure 4.14 shows the impact of different history sizes on the performance of BRE for three experimental runs, two during the day and one at night. It can be seen clearly that a threshold of 3 is indeed a sufficient value to discover active bursty links as it minimizes the transmission costs when compared to different history sizes as well as a standard routing protocol i.e. CTP. The results in Figure 4.14 also explain the variations in low-power wireless link qualities over time. Even the delivery cost of a routing protocol like CTP, which restricts communication only to consistently high-quality links, differs by one transmission per packet for back-to-back experimental runs. Nonetheless, the optimality of $n = 3$ holds true for all the three experiments.

Therefore, we classify every future packet - the short-term stability of a link - according to the recent success history of the link, i.e., whether the last n packet receptions

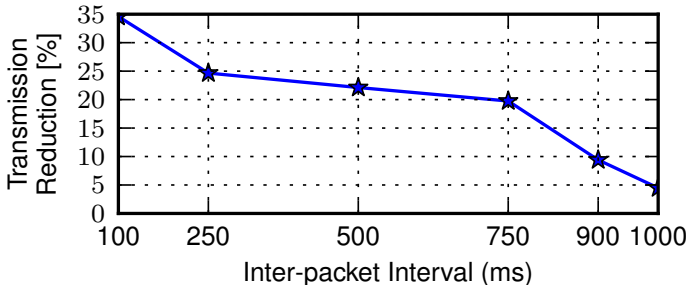


Figure 4.13 Impact of transmission speed on the performance of BRE for node-pair $9 \rightarrow 50$. With the increase in the inter-packet interval, the performance of BRE drops gradually. For the same node pair, the reduction in the number of transmissions drops from 34% at 100 ms to 4.9% at 1s.

were successful or not. In our prototype implementation, we used a threshold of 3, and the link estimator declared a link as active only after overhearing three consecutive transmissions from a particular sender-node. The results in the preceding sections prove the feasibility of this relatively simple approach.

Concluding, it is apparent to avoid using a very small reception *history* (i.e. $h = \{1, 2\}$) as it increases the total number of transmissions in the network. This is because a small *history* size is insufficient to predict the future quality of a link (cf. Section 3.4.2). Therefore, it can result in our link discovery mechanism being influenced by long unreliable links that rarely deliver a packet successfully. On the contrary, a very large history size (i.e., $h >> 3$) is also not feasible as it will subdue the use of bursty links and eventually catch up the conservative link selection approach based on average PRR, like in most of the current link estimation techniques.

To preserve the simplicity of our algorithm for our prototype implementation, we use a static threshold for declaring a link reliable for transmission. However, we believe that a more perceptive approach could also be useful. For example, one such approach would be to employ a learning phase at the start-up for calibrating the link estimator, such as in 4C [LC11]. Our major focus in this dissertation is to evaluate the feasibility of transmissions over bursty links, so we regard the investigation of such learning techniques as future work.

4.5.6 Overhead

We divide the overhead introduced by BRE into four different categories namely overhearing, processing, storage, and transmission. The passive overhearing technique employed by the link estimator comes at a cost because a node has to listen to the packets that are not addressed to it. However, due to the broadcast nature of wireless transmission, these packets are always received if the node's transceiver is in the receive state. State-of-the-art radio chips, such as the Texas Instrument's CC2420, can be configured to discard all the received packets that are not addressed to a node. Therefore, the overhead associated with overhearing amounts to packet reception and the processing required to deliver a packet from MAC to the link estimator.

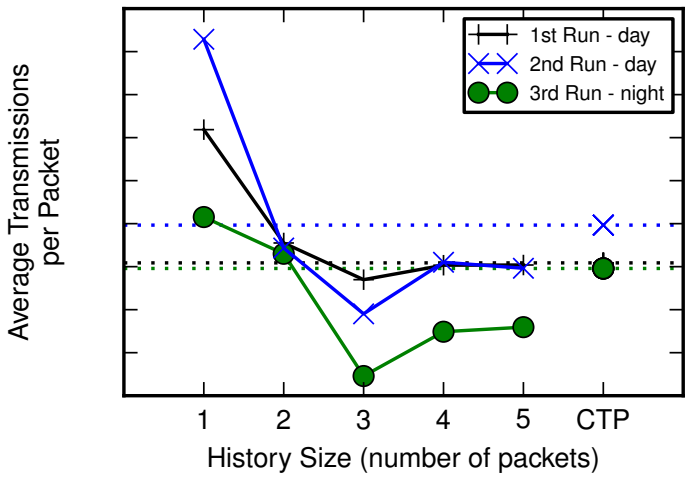


Figure 4.14 Evaluation of different history size thresholds for BLE on MoteLab. The dotted straight line represent the average of CTP for corresponding experimental runs. $h = 3$ performed the best overall.

Our current implementation of BRE requires 902 bytes of additional code memory and 270 bytes of additional data memory.

Finally, the only transmission overhead introduced by BRE is the announcement message sent by the overhearing-node to the sender-node informing about the temporary availability of a bursty link. There is no retransmission of this message because it also serves the purpose of testing the symmetry of that link. Moreover, as mentioned earlier, even a single successful transmission over a bursty link that reduces one hop would cancel out the overhead introduced by this additional message. However, Figure 4.12 shows that the burst lengths are much longer for most of our experiments. Considering the fact that BRE can reduce transmission costs by up to 40% and increase routing throughput by up to 20%, we believe that the processing, storage and transmission overhead, presented in this section, is reasonable.

4.6 Summary

We have presented a simple greedy approach (i.e., BRE) to utilize bursty links for packet forwarding during their stable transmission periods. BRE performs two main tasks: First, it integrates link estimators, such as BLE and STLE, into existing routing infrastructure. Second, it provides the corresponding routing extensions to utilize the links suggested by these link estimators. The ability of BRE to perform these tasks without introducing any changes in packet headers makes it transparent for the underlying routing protocols. Similarly, this feature also enhances the usability of BRE as its modular design is not tied to any specific routing protocol.

Our evaluation results show that by transmitting over long range intermediate links the number of transmissions in the network can be reduced. We observed that traditional routing protocols, such as CTP, even fail to utilize long-range bursty links with relatively longer transmission bursts. We also observed that BRE does not deteriorate the reliability of traditional routing protocols. BRE is particularly useful in networks with high density of nodes. This is because higher node density

increases the probability of finding a long-range intermediate link that offers a better alternative path than currently being used by the underlying routing protocol.

The data analysis presented in this chapter provides a greater depth of detail about the extent and applicability of the previously ignored class of links. We observed different characteristics of intermediate links such as timeliness and burst lengths. Another important finding is that the magnitude of improvement is higher for higher transmission rates. This is because at higher transmission rates wireless links show a strong correlation between packet reception events. Nonetheless, we believe that the average improvement of 19% over traditional routing by transmitting over links with high loss rates is a credible and a realistic result.

Overall, BRE is a lightweight approach for exploiting link dynamics in low power sensornets. However, it does not mean that BRE is not applicable in other classes of wireless networks such as meshnets. In Chapter 6, we show the generality of BRE by providing initial evaluation results from an IEEE 802.11 based testbed.

5

Probabilistic Addressing

In the previous chapters, we presented a detailed design and evaluation of our link estimator and routing extensions. However, our contribution so far has been focused on simple routing scenarios such as collection protocols in sensornets where the location of each participating node in a network is irrelevant. Each node in a network maintains a gradient towards the collection root. Similarly, every packet has only one possible destination, i.e., the root of the collection tree.

The integration of BLE and BRE in such *address-free* scenarios is straightforward since local changes in the gradient towards the root node do not impact the overall routing topology. However, this assumption is only valid in many-to-one scenarios. In point-to-point communication scenarios, such as in BVR, any temporary changes in the next hop selection may also change the routable address of each node on a particular branch of the routing tree. Hence, maintaining precise and up to date addresses of nodes in the network form the basis of point-to-point routing protocols. Typically, such routing protocols maintain an address database which has to be updated each time a node in the network changes its address. When a node wants to send a packet, it first has to query that database to get the routable address of the destination.

We now move on to the next problem space tackled in this dissertation, i.e., addressing. Our goal is to formulate an addressing scheme which allows the inclusion of intermediate links in routing process and thereby accepts rapid changes in routing paths. We aim at achieving this goal without compromising the stability of the addressing scheme.

The remainder of this chapter is structured as follows. In Section 5.1, we motivate the problem. Section 5.2 presents the background, introduces the design space and specifies target network types. We describe PAD's design and algorithmic details in Section 5.3. Section 5.4 thoroughly evaluates PAD regarding stability and adaptability. We present a simple routing strategy over PAD in Section 5.5. Results from our routing evaluation are discussed in Section 5.6. Finally, we discuss prominent related work in Section 5.7 before concluding the chapter in Section 5.8.

5.1 Motivation

Dealing with unreliable and highly dynamic wireless links is a major challenge in establishing stable addressing and routing in multihop wireless networks. It is further aggravated when location information is unavailable, a common situation in many wireless network deployments, as nodes have to determine their own addresses that reflect the underlying connectivity. As a result, rapidly changing link conditions do not only affect packet delivery and routing topology, but also the topological location and addresses of nodes.

A plethora of solutions [EFK07, FRZ+05, CA06, ERS06, MWQ+10, JS03, RRP+03, MOWW04] has been presented for situations, both in meshnets and sensornets, where location information is not available at the nodes and geographic methods cannot be used for routing. The majority of these location independent addressing and routing schemes are based on tree construction primitives: Ranging from simple data collection [GFJ+09] and dissemination [ERS06, LL08] to complex virtual coordinates based point-to-point routing [EFK07, FRZ+05] in meshnets and sensornets, tree-construction has established itself as common building block for location independent routing.

However, to ensure stable trees and addressing in the network, most tree-construction based addressing and routing schemes put excessive focus on tree maintenance and stability while adaptability gets compromised to a large extent [ALL+09, Vae10, AVL+11]. The underlying technique is to employ a long-term link estimator [FRZ+05] and select parent(s) only among neighbors with consistently high quality links. Although this results in consistent addressing and stable routing tree across the network, this long term binding restricts the network in how well it can adapt to link dynamics [SKAL08, ALL+09].

Similarly, tree based addressing and routing infrastructures suffer heavily from rapid topological changes due to varying link conditions in a network. Such situations often occur in a sparse network with a low density of nodes, where a node might have no reliable communication partner at all. In such situations we see frequent address changes and thus a significant overhead due to regular updates in the address database [VAW10, AVL+10, VW10]. Moreover, it would also result in inconsistent routing trees, introducing typical routing pathologies such as packet loss, loops, and stranded nodes.

5.1.1 Approach

In this chapter we show how to retain the benefits of tree based addressing and routing schemes without maintaining explicit trees in a network. The basic concept is the same: Determine a node's location based on the vector of hop counts from a set of landmarks[1] in a network. However, the execution of this concept is substantially different. In contrast to the existing approaches, our approach neither relies on long term link estimation nor maintains any explicit parent-child relationships in a network. Instead, this approach, named PAD, assigns probabilistic addresses to

[1] more often referred to as beacon nodes. We use the term landmark to distinguish it from *beacon* packets.

nodes. The basic idea is that a node learns from its past locations and calculates the probability distribution over its recent locations. This probability distribution is then used as address of the node. Hence, a node's location is defined in terms of the probability that it exists in a certain location and remains independent from the packet loss at shorter time scales. All other nodes in the network predict the current location of a node in its distribution. As a result, PAD decouples addressing from routing, allowing to adapt routing paths to the very recent network conditions. The design of PAD is inspired by *atomic orbitals* [Dai] that describe the probability of finding the electrons of an atom in specific regions. Thus, the location of an electron is defined in terms of the probability that it exists at a particular location around the nucleus of an atom. An alternative view on the PAD approach is that it uses fuzzy instead of sharp coordinates for nodes.

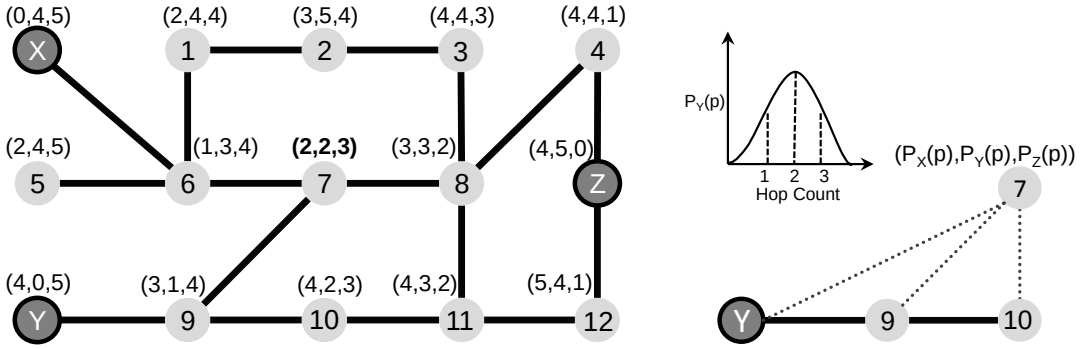
5.1.2 Major Contributions

This chapter makes the following key contributions:

- **Address Stability:** Compared to other addressing and routing schemes, PAD requires 3-7 times fewer address updates in a global location directory. At the same time, it maintains a small amount of state and requires considerably less effort and complexity in its mechanisms and implementation. We show that such stable addressing can be achieved even by considering only very recent link conditions instead of pessimistically overhearing and estimating links over a time period in the order of minutes (or hours).
- **Address Monotony:** Once an address update occurs, the difference between the old and new location of a node is 3-12 times smaller for PAD than for comparable approaches. This implies that the changes in PAD addresses are gradual, which helps routing success. Our evaluation shows that this phenomenon allows PAD to maintain more up-to-date yet stable node locations in the network and reduce packet loss.
- **Responsiveness:** By decoupling addressing from routing and link estimation, PAD can respond rapidly to changes in link quality which existing routing algorithms naturally avoid. As a result, each data packet can be forwarded on a different path depending upon the very recent network conditions. Our comparative analysis on three testbeds shows that even a simple routing strategy over PAD can reduce packet loss and minimize the number of transmissions required by a packet to reach its destination.

5.2 Preliminaries

Our key contribution is to enable stable addressing by introducing probabilistic addresses for wireless networks. Before delving into the technical details, we present the basic idea and put our contribution in a simple context. We also consider our target environments – data centric (sensornets) and ID centric (meshnets and MANETs) networks – and shed light on how these environments may benefit from the addressing mechanism presented in this chapter.



(a) Coordinate establishment in virtual coordinate based routing protocols.

(b) Probabilistic addressing: basic idea

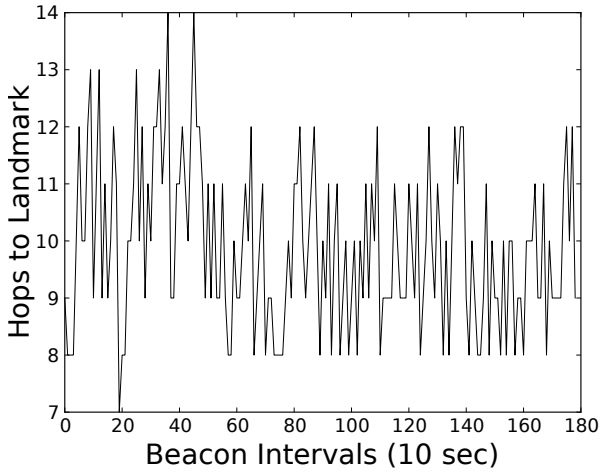
Figure 5.1 Addressing based on Virtual Coordinates

5.2.1 Basic Idea

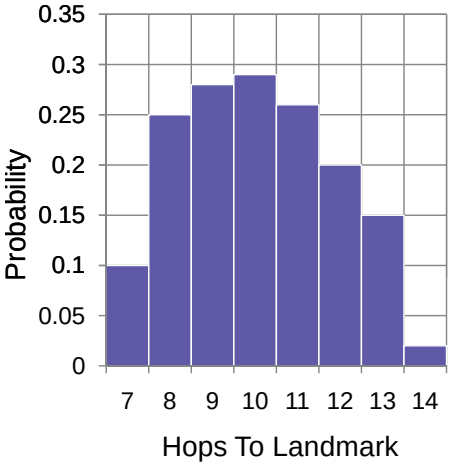
To establish a basic understanding of the design space we are working on in this dissertation, consider Figure 5.1(b), which represents the lower left segment of the network in Figure 5.1(a). The basic design philosophy of tree based routing protocols restricts nodes to select a single parent for each landmark and define their coordinates based on the hop counts achieved over these parents. As a result, a main challenge in tree construction based routing is that the changes at one node induce changes in all child nodes further down the tree. For example, node 7's virtual location with respect to landmark Y will heavily rely on the path $7 \rightarrow 9 \rightarrow Y$. Each time a node changes its hop distance from a landmark, all child nodes have to modify their hop distances to that landmark as well. As a result, any node failure or changes in the quality of the links (due to data loss) on this path will not only trigger a change in the routing topology but also in the virtual coordinates (location in the network) of node 7. To cope with this challenge, maintaining trees and virtual coordinates across the network which are particularly consistent is understandably the main objective of tree-based routing protocols. Therefore, they willingly concede performance penalties to achieve this objective.

In this dissertation, we oppose this philosophy and propose to break the stringent parent-child relationship. For example, let's suppose that node 7 can also reach landmark Y over the unreliable paths $7 \rightarrow Y$ and $7 \rightarrow 10 \rightarrow 9 \rightarrow Y$ (as shown in Figure 5.1(b)). We define a node's location on the basis of all possible paths that can be used to reach the landmarks regardless of the estimated quality of these paths, just like an orbital function describes possible quantum states of an electron around an atom [Dai]. We expect unstable coordinates based on these paths to exhibit a quantifiable, stable pattern. Nodes can keep track of the changes in their own coordinates and learn the associated patterns over time. Figure 5.2(a) depicts such a scenario from a real testbed by showing the development of a node's distance from a landmark over time. A similar argument can be made for cases where a node has no reliable neighbor over longer periods of time. In such a dynamic[2] network, assigning

[2]We only employ the dynamics that occur due to frequently changing link qualities and node failures.



(a) Development of hop distance from a landmark over a period of 30 minutes.



(b) Distribution of hop distance from a landmark.

Figure 5.2 In a pathological case, a node's distance from a landmark can vary significantly over time. A link estimator is typically used to filter out such dynamics. In sparse networks with challenging link conditions, even link estimators would struggle to maintain a stable routing topology. Assigning static virtual coordinates in such dynamic situations often results in unstable addressing.

static addresses to nodes often results in inconsistent trees, because a node's distance from the landmarks changes rapidly.

The central idea is to locate and address a node using these patterns instead of its absolute, current coordinates. Our addressing mechanism, i.e. PAD, therefore addresses a node in the form of a probability distribution (see Figure 5.1(b) and 5.2(b)) instead of a static location. Other nodes can then use this probability distribution as the destination address for packets to this node. Overall, PAD decouples addressing from routing and exposes multiple locations and paths to a node. This gives routing protocols the flexibility to exploit interesting communication opportunities on short-term stable paths towards the destination. For example, by employing BRE-like routing strategy without worrying about the stability of the addressing scheme.

5.2.2 Target Networks

In this section, we investigate the target networks that can benefit from the approach presented in this dissertation.

5.2.2.1 Sensornets

For a long time, routing in sensornets was limited to simple collection and dissemination primitives that do not require to reach a specific node based on its identifier. However, a vast majority of current applications in sensornets – such

as data centric storage [ERS06], data query methods [GEH03, LKGH03], pursuer-evader games [DAG03], industrial automation, light control networks and cyber physical systems in particular demand individual addressing support. Approaches like BCP [MSKG10] and BRE [ALL+09] provide mechanisms to exploit path diversity and link dynamics in sensornets but do not consider addressing. Moreover, the networking conditions reported in the literature [SPMC04, WWAL+05, RL09] are very challenging in sensornets mainly due to two reasons: (1) harsh environmental conditions from a networking point of view, and (2) a rapidly changing network topology resulting from frequent node failures.

5.2.2.2 Manets and Meshnets

Manets and meshnets are address centric, i.e., here the goal is to assign a unique identifier to each node and share resources among the participants, such as an Internet connection. The presence of intermediate and bursty links has been recurrently reported in the literature [SKAL08, ABB+04]. Such networks also present challenging conditions due to interference from other coexisting networks on the same frequency band and due to rapidly growing and shrinking numbers of participants. Opportunistic routing [BM05a] provides an elegant solution to exploit path diversity in such networks. We share the same spirit, but differ significantly in detail. Moreover, opportunistic routing neither deals with addressing, as it operates on fixed geographic locations and IP addresses, nor focuses on challenging conditions in the network and their corresponding impact on addressing.

Our discussion in the remainder of this chapter focuses on sensornets. In Chapter 6 we discuss our results from meshnets including mobile nodes.

5.3 Probabilistic Addressing Explained

Approaches such as BVR attempt to filter out the variability in a node's coordinates, which is caused by network dynamics, to obtain a stable address. In contrast, PAD incorporates this variability into a node's address by encoding a limited history of the node's varying coordinates. The idea is to learn from the dynamics exposed by a node's virtual coordinates and express them in the form of probabilities. The routing algorithm can then determine a node's coordinates by predicting its current location in its probability distribution.

To arrive at a stable address, a PAD-node needs to iteratively (1) collect its coordinate history, (2) calculate and encode its address, and (3) disseminate its coordinate and addressing information to its neighbors via beacons. The following sections explain these steps in detail.

5.3.1 Coordinate History

PAD-nodes determine their network coordinates based on the beaconing mechanisms of established virtual coordinate systems [FRZ+05, CA06]. Thus, in a network with λ landmarks, the node S has the coordinates $\vec{c}(S) =< h(S, L_1), \ldots, h(S, L_\lambda) >$ as

discussed in Section 2.3.2.1. Note that such coordinates reflect the current shortest paths between S and each of the landmarks without any further filtering, link estimation, or quality information.

All nodes in the network determine their coordinates periodically, once per beacon interval β. In PAD, each node S collects its σ most recent coordinates in its coordinate history, which is a table of size σ comprising of coordinate vectors:

$$\mathcal{H}(S) = \begin{pmatrix} \vec{c_1}(S) \\ \vdots \\ \vec{c_\sigma}(S) \end{pmatrix} = \begin{pmatrix} < h_1(S, L_1) & \dots & h_1(S, L_\lambda) > \\ \vdots & \dots & \vdots \\ < h_\sigma(S, L_1) & \dots & h_\sigma(S, L_\lambda) > \end{pmatrix}$$

In each beacon interval, S updates this history by adding its latest coordinates and – if necessary to maintain the maximum history size σ – by evicting the oldest coordinate vector.

5.3.2 Address Calculation

After updating its coordinate history, node S re-calculates its address as follows. For each landmark L_i (i.e. each column in $\mathcal{H}(S)$), it determines which hop-count values to L_i the history contains and how often they occur. In other words, it calculates the frequency distribution of unique hop counts for landmark L_i as the set of tuples:

$$\mathcal{F}_i(S) = \{(h_1(S, L_i), f_1), \dots, (h_\delta(S, L_i), f_\delta)\}$$

where the tuple $(h_j(S, L_i), f_j)$ consists of the unique hop-count value $h_j(S, L_i)$ and its absolute frequency (or number of occurences) f_j in $\mathcal{H}(S)$.

For example in an increasingly unstable network, the number δ of tuples in $\mathcal{F}_i(S)$ would grow as the shortest paths from S to L_i increasingly vary in their length. At the same time, the absolute frequencies f_j of each of the hop-count values would decrease as their sum, by construction, cannot exceed σ. In a very stable network, however, the history would report hop counts for only one or a few shortest paths between S and L_i, so $\mathcal{F}_i(S)$ would contain only one or a few elements which would have large absolute frequencies.

After node S determined the frequency distributions $\mathcal{F}_i(S)$ for all landmarks, it constructs its routable address simply as the vector:

$$\vec{a}(S) = < \mathcal{F}_1(S), \dots, \mathcal{F}_\lambda(S) >$$

With this information, the probability distribution of a node's coordinates can be readily derived from its address. Thus, a PAD address contains a notion of path quality reflected by the number of different paths and their variance in length. Our results in Section 5.4 indicate that, given a suitable history size σ, the set of frequency distributions in an address stabilizes in the long run and is largely unaffected by short-term link conditions. Furthermore, we empirically observed that the frequency distributions contain only a small number of unique hop counts, even under challenging link conditions. Thus, it is more efficient to encode them as variably sized sets than, e.g. as fixed-size arrays similar to the representation of the coordinate history.

After calculating its new address $\vec{a}(S)$, S compares it to its previous address $\vec{a}'(S)$ by calculating a difference value d (e.g. via Pearson's χ^2-test). If d exceeds the threshold ϵ, S needs to update its address in the address database[3] of the network. Since an address update is an expensive operation, it is important to choose the threshold ϵ appropriately as discussed in the evaluation in Section 5.4.

Note that the node coordinates $\vec{c}(S)$ are deliberately based on the minimum hop distance metric $h(S, L_i)$ in PAD because: (1) it is a simple and established selection criterion, (2) it simplifies efficient routing and helps to avoid loops, and (3) it helps to keep the number of paths represented in PAD-addresses low.

5.3.3 Address Dissemination

Our approach is based on periodic beacon exchanges among neighbors. It is not bound to any specific beacon exchange rate or technique and, in principle, it should work with any technique presented in the literature, such as adaptive beaconing [GFJ+09].

In PAD, each node S broadcasts a beacon packet once per beacon interval with the following information:

- **Node Coordinates:** The current vector $\vec{c}(S)$ of minimum hop distances $h(S, L_i)$ from S to each landmark L_i. To reemphasize, this does not use or contain link quality information.
- **Node Address:** The current address $\vec{a}(S)$ of S, i.e. the frequency distribution of its coordinate history. Including the address in beacons is a tradeoff of communication overhead over computational and space overhead. Alternatively, any neighbor T of S could record and maintain a history of S's coordinates itself to compute the address of S (cf. Section 5.3.2).
- **Traces**: For each landmark L, a trace of the last 5 nodes on the path from L to S. This back-tracking information helps to prevent loops in the paths to a landmark.
- **Neighbors**: A list of neighbors from which S received a beacon in the last beacon interval. This piece of information is used to identify neighbors with symmetric links for routing. This mechanism is similar to the regular exchange of reverse ETX messages for each neighboring link as used by current routing protocols.
- **Sender ID:** The unique ID[4] of S.
- **Sequence Number:** A sequence number for the beacon packet assigned by S.

The size of beacons depends on the number λ of landmarks in the network and the number of *symmetric neighbors*. We defer a more detailed discussion on memory and beacon size tradeoffs to Section 5.6.4.

[3]Location services for PAD are out of scope in this dissertation. There are well established location services for virtual coordinate based routing protocols, such as [OBM+07, CM02].

[4]We distinguish the term *ID* from *address*. Each node in the network has a unique and immutable ID in the network while an address is a node's relative location in the network used for routing.

5.3.3.1 Hidden Loop Avoidance

Loops (or cycles) occur due to sudden changes in the routing topology and have been thoroughly addressed in the literature [GFJ+09, EFK07]. While they manifest in routing, the dynamics of PAD have the potential to introduce hidden loops in its addresses. Therefore, we tackle this issue during address establishment.

We employ a very simple mechanism to avoid hidden loops in the PAD addresses of each node rather than detecting them. Every beacon is appended with a trace of the last τ nodes in the reverse path (from the sender node towards the landmark) that it offers. The idea is that a node S will check for its ID in each trace before including the corresponding path in its coordinate $\vec{c}(S)$. Ideally, such a trace should include all the nodes on the path, however, this would have severe implications on the scalability of PAD. From our empirical data, we found out that a value of $\tau = 5$ is a sufficient value[5] to avoid loops. First, because we always select the smallest hop distance to landmarks for deriving a node's coordinates. Second, because such loops mostly occur in the local vicinity of a node (two to three hop neighborhood). We employ this solution for simplicity in our prototype implementation of PAD. Exploring other scalable approaches to avoid hidden loops [EFK07] is a future work.

5.3.4 Summary

Overall, this design is inspired by the notion of electrons surrounding an atom on an orbital cloud where the position of an electron can be predicted with a certain probability. Similarly, we use a probability function to predict the position of a node – and hence its address – at a specific point in time. As a result, PAD decouples addressing from routing to provide a consistent routing topology even in the existence of links with highly variable qualities. Moreover, PAD encodes the quality of multiple paths leading to a node into the address distribution of a node. As result, a node can make forwarding decisions dynamically on a per-packet basis based on this path set.

Routing on top of such an address distribution offers a number of decision choices in terms of routing metrics. For example, a straight forward metric is to rely on the coordinate mean to address a node and to make forwarding decisions. We discuss this routing metric, advanced metrics, and optimizations in Section 5.5 after detailing on the system design and evaluating address stability.

Before concluding the design of PAD, we revisit our claims and point out how we achieve them in our design to facilitate a complete understanding of the system. First, we do not maintain any long-term parent-child relationship in PAD. A node's coordinates are based on a single path and include the notion of a single parent. However, these coordinates are just part of the mechanism to derive the PAD addresses, which themselves do not maintain any parents in the network. Moreover, PAD also allows to decouple addressing from routing as each node has an independent location and is not part of any routing tree. Specifically, any change in the route from a landmark to a node, and vice versa, would not change the address of

[5] This assumption is at least true for the three testbeds we have used in our experiments. The maximum path length in our experiments is 10 nodes.

Testbed	Available Nodes	Average Node Degree	Tx Power Level Used
MoteLab	99	7.2	−15 dBm
Indriya	125	18.5	0 dBm
Twist	94	23.3	−15 dBm

Table 5.1 Basic characteristics of the three testbed we used in our experiments. All these testbeds are comprised of IEEE 802.15.4-based TMote Sky nodes. Node degrees, i.e. average number of one hop neighbors, were derived for the respective transmission power levels.

that node. Finally, our approach is based on a typical, low-rate broadcast-beaconing mechanism employed by existing routing protocols and does not rely on expensive overhearing of data transmissions in the network.

5.4 Performance Evaluation of PAD

Our evaluation of PAD focuses on two aspects: (1) We need to choose an appropriate history size σ and error level ϵ between PAD coordinate distributions (cf. Section 5.3.3). σ is the size of the history of node coordinates, while ϵ is the threshold for deciding whether the differences between a newly calculated PAD address and the previous one are significant and hence require an update in the global address database. (2) We need to thoroughly compare PAD with existing virtual coordinate based addressing approaches to observe potential benefits and drawbacks of our approach. PAD is implemented for TinyOS 2.1, and has been tested in the TOSSIM simulator [LLWC03,LAW08] and on IEEE 802.15.4-based Tmote Sky platforms. We compare PAD with BVR [FRZ+05], the only implementation of a virtual coordinate based routing protocol available for a realistic comparison.

We first briefly discuss our experimental setup and the testbeds we used in our experiments.

5.4.1 Testbeds and Experimental Setup

Evaluation on real testbeds is mandatory to explore the efficacy of the concept presented in this chapter. We utilized three widely used IEEE 802.15.4 based testbed deployments for our evaluation, namely MoteLab [WASW05], Indriya [DCA09], and TWIST [HKWW06]. All three testbeds are indoor deployments – nodes are deployed on multiple floors of buildings – with coexisting IEEE 802.11 deployments. We used different transmission power levels to stress-test PAD under varying network conditions and topological characteristics. The major characteristics[6] of these testbeds are shown in Table 5.4.

- **MoteLab** is a 184 node deployment on three different floors. Among the three testbeds, MoteLab is the sparsest deployment – only 93 nodes were

[6] Although we discussed these testbeds in Chapter 4, we have to present their characteristics again since our experiments span a large period of time during which their characteristics changed significantly.

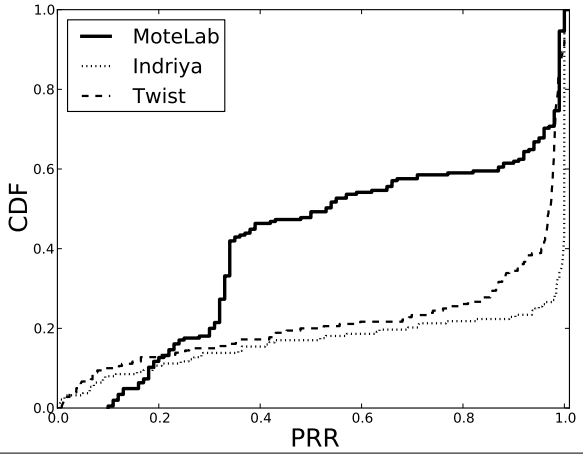


Figure 5.3 CDF of link qualities measured on the three testbeds. Almost 60% of the links in MoteLab have PRR's below 0.8 compared to just 20% of such links on Indriya and TWIST. We only include links on which at least 10 packets were received.

active during our tests – with an average node degree of 7. MoteLab serves as a sanity check for PAD evaluation as it presents very challenging network conditions (see Figure 5.3).

- **Indriya** is a 127 node deployment on three different floors. The network topology of Indriya is very similar to MoteLab, however, the overall network connectivity in Indriya is better than in MoteLab. 125 nodes were available to us for experiment. We reduce the transmission power to $-25\,\mathrm{dBm}$ to increase the network's diameter.

- **TWIST** is a 100 node deployment (94 available). TWIST is the densest deployment among the three, and path lengths are quite small: Most of the nodes can reach each other directly when transmitting at full transmission power. Therefore, to create a multihop network we reduce the transmission power to $-25\,\mathrm{dBm}$.

The major characteristics of these testbeds are shown in Table 5.4. Figure 5.3 shows the CDFs of link qualities on all the three testbeds and clearly points to the challenging nature of MoteLab: Almost 60% of the links have PRRs below 0.8 compared to just 20% of such links on Indriya and TWIST. The outcome of Figure 5.3 is essential for understanding the results in Section 5.6.

5.4.2 Determining the System Parameters

Before evaluating the stability of addresses in PAD and comparing it to related approaches, we need to calibrate the core parameters of our system: the history size σ and the error probability ϵ. Although both σ and ϵ are user-desired accuracy thresholds, we derive their values here for completeness and for evaluation purposes.

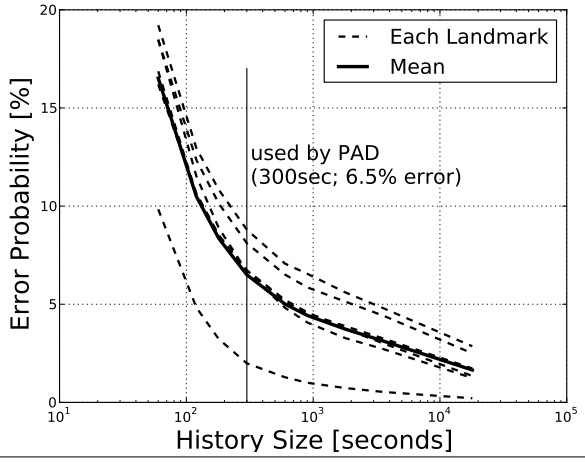


Figure 5.4 Pearson's χ^2-Test for deriving history size σ and error ϵ: The graph shows a gradual decrease in the error probability for smaller history sizes. PAD uses the cutoff at $\sigma = 300$ sec and $\epsilon = 6.5\%$, as beyond that point only a slight decrease in the error probability introduces significant memory overhead and impedes the adaptability of addressing. (n.b. log scale on x-axis)

5.4.2.1 History Size

Our first evaluation factor is to determine the appropriate sample size for the probabilistic addressing, i.e. the coordinate history size σ (see Section 5.3.1) that shall be used to calculate the PAD addresses. The goal is to strike a suitable tradeoff between the stability and adaptability of PAD. It means that we need to find the minimum sample size (coordinate history size σ) that results (within an acceptable error threshold ϵ) in a stable distribution. In order to find this we use Pearson's χ^2-Test. It is a test of goodness of fit, which derives how much two distributions differ from one another. Our goal is to calculate a $p - value$ (error probability) that reflects how likely it is that the differences between two distributions are caused by chance.

To perform this analysis we ran PAD with six landmarks on the TWIST testbed at a transmission power level of -15 dBm. Each node generated a beacon every 10 seconds[7] for a total runtime of 24 hours. Our reference distribution of each node's coordinates for the χ^2-Test is derived from the whole data set of 24 hours. Our actual coordinate distributions are comprised of smaller segments of the whole experiment duration increasing in size. The smallest time frame we compare the distributions for is 60 seconds. We compare these distributions with the reference distribution to derive a minimum history size. The goal is to find the smallest history size for a node's coordinate distribution than can accurately represent the reference distribution of the whole experiment duration.

Figure 5.4 shows the average[8] p-value for different history sizes. It shows that there is a rapid decrease in the error probability for smaller history sizes. However, later increasing the history size does not substantially impact the error probability any-

[7]The choice of the sending rate over a longer period of time is irrelevant here. We wanted to collect maximum data without saturating the network to derive a stable reference distribution of the coordinates.

[8]We averaged the p-values of the corresponding distributions of a particular history size over all landmarks.

more. For example, when increasing the history size from 60 to 300 seconds, the error probability decreases from 17% to 6.5%. Thereafter, increasing the history size from 300 seconds to 1000 seconds only results in a 2% decrease in error while significantly dampening the adaptability of the coordinates and increasing the memory overhead due to the larger history size required to compute the PAD address. Here we can tradeoff a slight inaccuracy for a higher adaptability and smaller memory overhead.

The cutoff used in PAD is therefore at a history size σ of 30 beacons, i.e. 300 seconds in this case. For the remaining evaluation in this chapter, we calculate PAD addresses from a history comprising the last 30 beacons. Our results indicate that even with this history size PAD achieves at least three times more stable addressing than BVR. This result roots in the fact that BVR itself trades stability for adaptability by employing a highly pessimistic and cautious approach for changing the address of a node.

5.4.2.2 Error Threshold

The error threshold ϵ is the threshold for deciding whether the differences between a newly calculated PAD address and the previous one are significant and hence require an update in the global address database: After calculating its new address $\vec{a}(S)$, a node S compares it to its previous address $\vec{a}'(S)$. If the difference exceeds the threshold ϵ, S needs to update its address in the address database. Hence, ϵ allows the user to tradeoff address updates for routing inaccuracies. We evaluated the stability of PAD with different ϵ values and observed that for smaller values – ranging from 1% to 10% – ϵ does not impact the rate of address updates in the network. In our evaluation we use $\epsilon = 6.5\%$ as a representative value within that range.

Although both ϵ and σ thresholds are empirically derived from testbed results, self-calibration of these thresholds would be a preferred solution such that the network would optimize them according to the observed conditions. Nonetheless, any such self-calibration mechanism requires additional memory and computational overhead (e.g. to store, calculate and compare reference distributions of coordinates), which is not desirable in sensornets.

5.4.3 Comparison with BVR and S4

Deriving the error threshold ϵ and the history size σ completes all the pieces of our design. Now we thoroughly compare PAD with the addressing mechanism of BVR. We defer the discussion on routing over PAD to Section 5.5.

Although S4 is considered state-of-the-art in sensornets, our evaluation in this section only compares PAD with BVR. This is because S4 extends BVR with its cluster based routing approach to guarantee reachability at the cost of relatively higher state – maintaining both local-cluster and global states. Whereas, the establishment of *global* coordinates in an S4 network is exactly based on BVR: S4 even uses the code base of BVR. Thus our evaluation in this section accounts for both S4 and BVR with one exception: Routing in S4 is based on the closest landmark to the destination,

and hence, updates in the address database are only needed when a node's closest landmark changes. BVR on the other hand requires an address update for every change in any of the coordinate components, because it greedily routes a packet based on the address vector of the k closest landmarks. The rate of coordinate change in S4 would still be the same as for BVR. Moreover, because S4 uses hop-count as its performance metric, later in Section 5.6 we show that PAD outperforms S4 when the comparison is performed on a testbed and is based on a prevalent metric, i.e. total *number of transmissions* in the network.

We use the latest releases of BVR and S4 from the TinyOS code repository[9]. To ensure that our results are not unjustifiably influenced by the state-of-the-art implementation of the TinyOS 2.1 [Dev08] communication stack, we had to update the APIs of S4 and BVR. However, these changes are only minimal and correspond to slight changes in platform independent APIs, such as *send* and *receive*. We did not alter any other parameter or algorithmic aspect of the protocol itself.

Our comparison with BVR is based on four factors.

- **Address Stability:** To compare the rate of changes in the addresses in PAD and BVR. It is defined as the share of beacon intervals in which the nodes changed their addresses. This is our key evaluation aspect to show the stability of addresses over time.
- **Address Monotony:** To measure the difference between hop distances from landmarks over time. This analysis looks at each component of the address vector to analyze the range of change in hop distances from each landmark.
- **Hop Distance:** To measure the hop distances from landmarks. The goal of this analysis is to see the average hop distance achieved by both addressing techniques.
- **Node Dynamics:** To observe the stability of PAD addressing with regard to node dynamics, i.e. frequent node additions and failures in the network. This analysis will give us hints about how well PAD recovers from such dynamics in the network.

Our experiments for this analysis share the following common characteristics for PAD and BVR: (1) Each experiment starts with an initial calibration phase during which each node transmits a beacon every second. The goal is to allow BVR to stabilize its link estimates and thereby its virtual coordinate system. However, at any instant, PAD's address distributions are always derived from a history of the last 30 beacons. (2) After the calibration phase the evaluation phase starts in which each node transmits a beacon every 10 seconds[10] on Motelab and Indriya. (4) We do not restrict the neighbor table size and therefore each node is allowed to maintain the state of all its one-hop neighbors with symmetric links. (5) On each

[9]These releases are compatible with TinyOS 2.0, but there are significant differences, e.g. the communication stack, device drivers and interfaces, between the current release TinyOS 2.1 and the first release of TinyOS 2.x.

[10]The relatively faster beaconing is used to increase our data set on MoteLab and Indriya, as both these testbeds limit the time period of experiment runs.

testbed we preconfigure six well spread nodes to act as landmarks. The landmark selection is a well studied research area [CA06, OBM+07] and is not our focus in this disseration. (6) Finally, the link estimator in BVR employs passive overhearing of all transmissions in the network. This mechanism is not used in PAD.

5.4.3.1 Address Stability

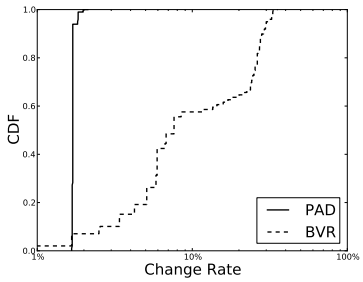
Address stability is an important factor in a virtual coordinate based routing infrastructure, especially for applications that cannot maintain the state of each node in the network and require a lookup mechanism in some location database. First, because routing to outdated addresses leads to routing failures. Second, because rapid changes in the addresses create heavy update and lookup traffic overhead that can be detrimental for network performance, especially close to the nodes responsible for maintaining the address database. Figures 5.5(a), 5.5(b), and 5.5(c) show the cumulative distribution of the nodes' change rate in terms of percentage of the beacon intervals in which the nodes change their addresses. By employing this metric we can assume that the rate of sending beacons does not impact the change rate of addresses, since sending beacons at higher rates increases the chance for changes in addresses in a certain time period but also increases the total number of beacons by the same amount. The CDFs clearly indicates that PAD's addresses are significantly more stable than BVR's.

Figure 5.6 shows the address change rate for each node in all three testbeds. It can be seen that PAD addresses are quite stable even under challenging network conditions (such as in MoteLab), where BVR's addresses have significantly higher change rates. From this analysis we can conclude that instantaneous changes in link conditions may lead to coordinate changes in the addressing mechanisms that assign static virtual coordinates to nodes at any instant. However, the underlying patterns of these instantaneous changes show stable distributions even at a time scale as short as 300 seconds.

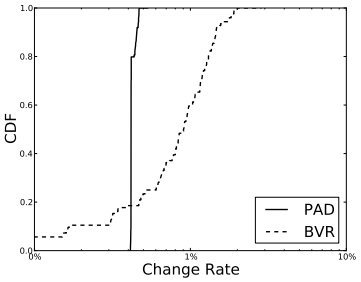
5.4.3.2 Address Monotony: Magnitude of Change

The magnitude of change determines the difference between a node's virtual addresses. For example, if a node changes its hop distance from a landmark from 4 to 6, its magnitude of change (or range) would be 2. The magnitude of change in a node's address is calculated by summing up the magnitude of change in each address-vector. Figures 5.5(d), 5.5(e), and 5.5(f) show that the magnitude of change in addresses is significantly smaller in the case of PAD (see Figure 5.6 for magnitude of change in each node's address).

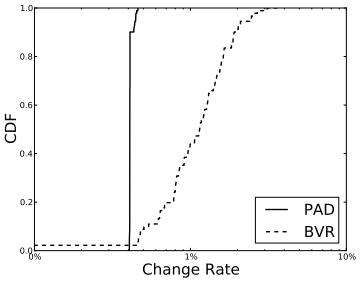
PAD shows a smaller magnitude of change because it sticks to the minimum hop distance path towards the landmarks. However, in BVR the address changes are more influenced by the ETX metric that favors long term stable links to achieve stable addressing in the network. As a result, the magnitude of change in addresses can be significantly higher. For example, BVR may select a longer but stable path after the previous path became invalid, whereas PAD's distribution will always favor the path with smallest hop count (see Section 5.3.2).



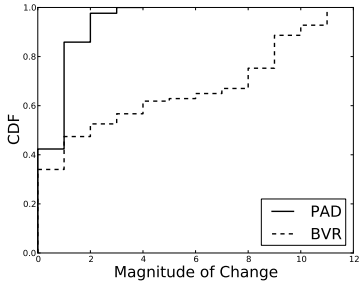
(a) Change Rate (Share of Intervals with Change, %) in MoteLab



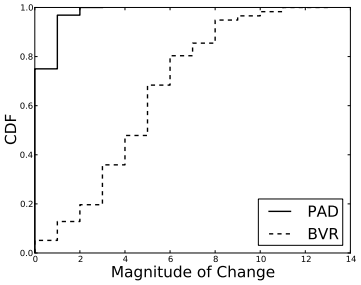
(b) Change Rate (Share of Intervals with Change, %) in Indriya



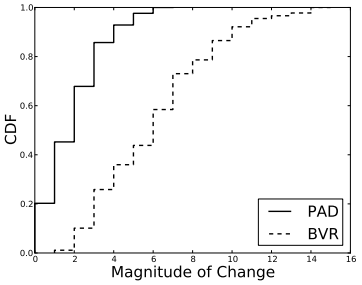
(c) Change Rate (Share of Intervals with Change, %) in Twist



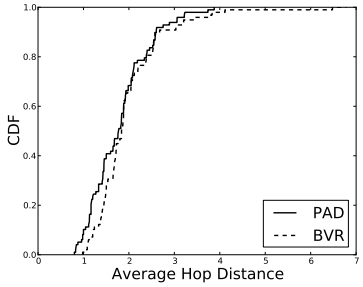
(d) Magnitude of Change in MoteLab



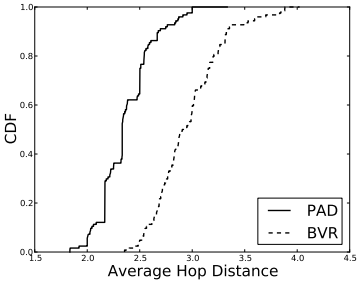
(e) Magnitude of Change in Indriya



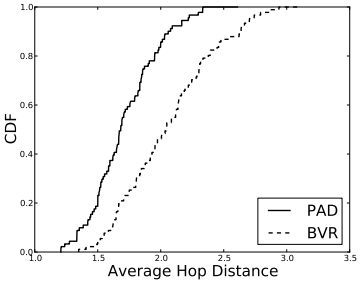
(f) Magnitude of Change in Twist



(g) Average Hop Distance from Landmarks in MoteLab



(h) Average Hop Distance from Landmarks in Indriya



(i) Average Hop Distance from Landmarks in Twist

Figure 5.5 Results from the address stability comparison: The CDFs of our three evaluation factors from three testbeds indicate that PAD reduces the rate of change in addresses, minimizes the hop distance from landmarks, and decreases the magnitude of change in addresses on all testbeds.

The smaller range of addresses implies that the changes in PAD addresses are gradual and a node's virtual location differs only minimally. Therefore, the packet routed towards a certain node is still routed to the vicinity and has a higher probability of reaching the target node even if that has changed its original address. However, in BVR routing to outdated addresses may lead to routing failures due to nodes taking significantly different virtual locations. Our evaluation in Section 5.6.3 supports this claim by showing that PAD reduces packet loss associated with address changes when compared to BVR.

5.4.3.3 Hop Distance

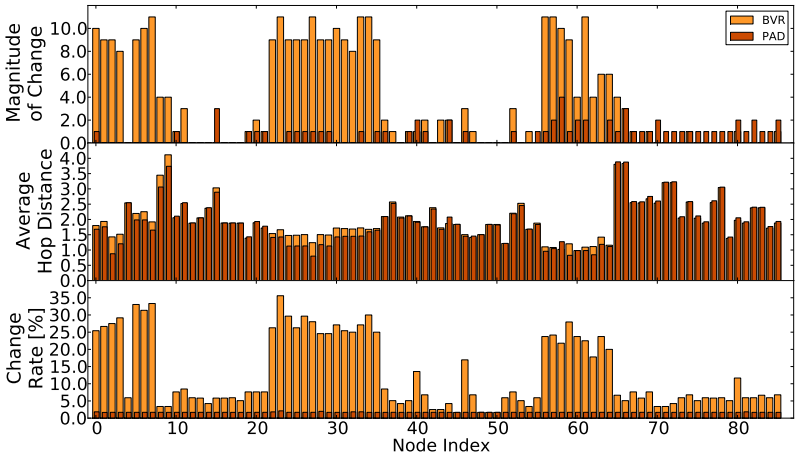
Figures 5.5(g), 5.5(h), and 5.5(i) depict the hop distance averaged over all landmarks. It can be seen that PAD achieves a significantly lower *mean* hop distance than BVR. Figure 5.6 also illustrates the per-node average hop distance. On all three testbeds, PAD achieves a smaller hop distance to landmarks. As discussed in Section 5.3.2, this is due to PAD preferring short paths to dominate its coordinate distributions.

We can draw two conclusions from these results:

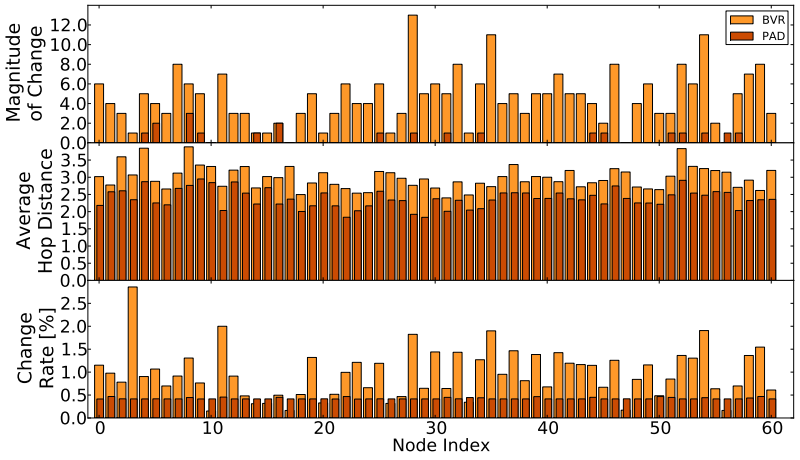
- The smaller hop distance means that the *mean* virtual coordinates (see Section 5.5.1) derived from PAD's distributions are smaller than BVR's virtual coordinates, and it may lead to nodes taking similar *mean* coordinates in dense networks. However, this is only true for the *mean* coordinates, whereas the actual PAD addresses, i.e. the coordinate distributions, shall always result in more diverse coordinates than in BVR.
- PAD's *mean* coordinates reduce the overall distance of nodes from landmarks (i.e. the depth of the tree in conventional approaches). It means that routing towards or from a landmark could reduce the hop distance and the number of transmissions required by a packet, if we can accurately predict the fate of the transmissions on shorter but more unreliable paths (see Section 5.6 for detailed results). For example, by employing BLE as a link estimator.

5.4.3.4 Node Dynamics

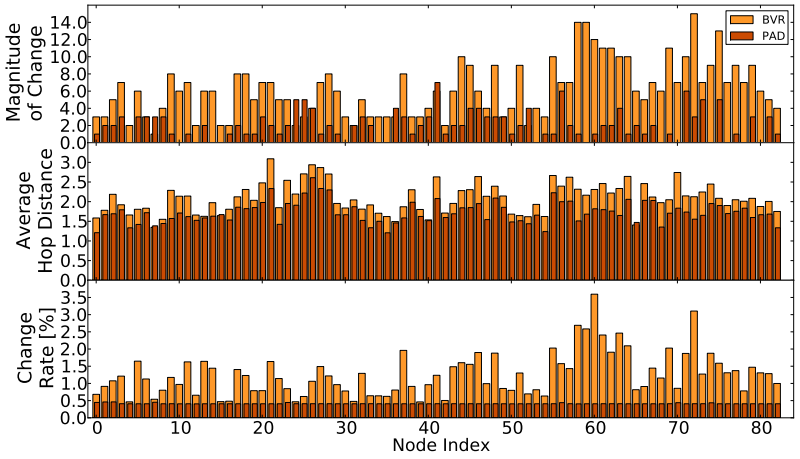
After evaluating the stability of PAD under different network conditions, we now evaluate PAD from another perspective, i.e. by growing and shrinking the size of the network to see how well PAD integrates additional nodes and recovers from node failures in the network. We use the TOSSIM simulator to introduce such node dynamics in the simulated network. We use a 100-node grid-like topology in TOSSIM with four nodes configured as landmarks. Our first experiment starts with 50 nodes, and 10 new nodes are added to the topology after every ten minutes. Similarly, our second experiment starts with 100 nodes, and 10 nodes are deleted from the topology after every ten minutes. Figure 5.7 shows our results where each data point represents the addition (see Figure 5.7(a)) or deletion (see Figure 5.7(b)) of 10 nodes. We can clearly see that PAD achieves far less address changes in the network than BVR. This is because PAD stabilizes quickly and an addition of a new node only affects PAD's addresses if it offers a smaller hop distance than the ones



(a) MoteLab

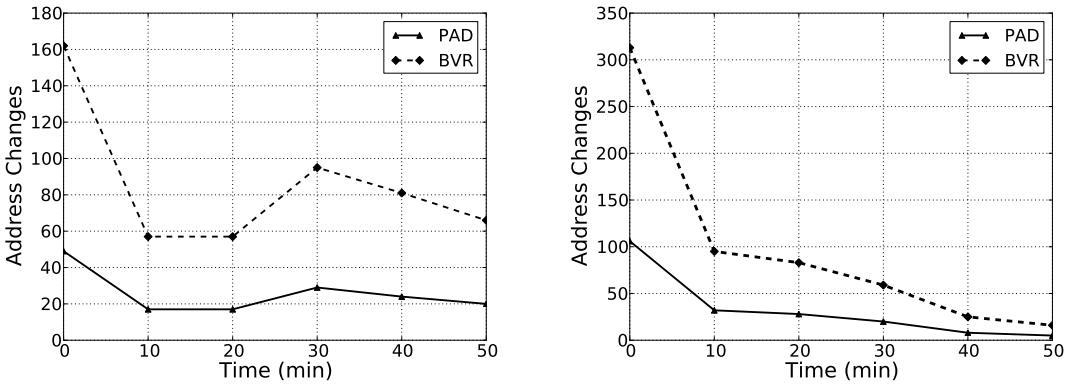


(b) Indriya



(c) Twist

Figure 5.6 Per-node analysis for the three testbeds. The results show significant improvements even under challenging network conditions as experienced in MoteLab. The figures only show the data for the nodes that were available for all our experiments as different nodes failed and were repaired throughout the time period of our experiments.



(a) Node Addition (b) Node Deletion

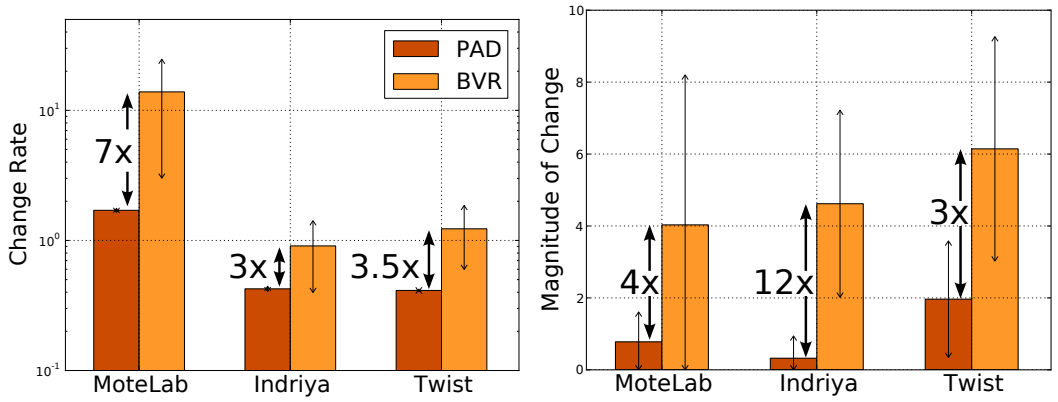
Figure 5.7 Node Dynamics: PAD achieves significantly fewer address changes in the network due to node dynamics. Each data point represents adding or deleting 10 nodes from the network. In total, PAD results in 154 and 201 address changes compared to BVR's 508 and 593 changes due to node addition and deletion, respectively.

reflected in PAD distribution. However, link estimation based addressing in BVR takes time to incorporate new nodes and stabilize its link-metric and addressing across the network. Hence, these results prove the flexibility of PAD for networks with a rapidly growing and shrinking number of participants.

5.4.3.5 Summary and Comparison with CTP

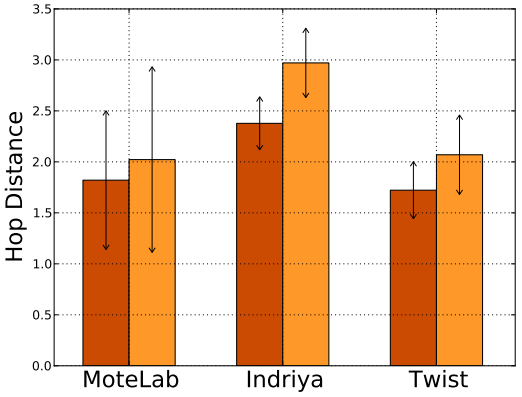
Figure 5.8 summarizes our results regarding address stability. PAD achieves 3 to 7 times more stable addressing than BVR on Indriya and MoteLab, respectively. An alternative way to formulate these results would be that BVR achieves 89% stability and PAD achieves 98.5% stability on MoteLab: In every 1000 beacon intervals, a node changes its address 110 times in the case of BVR and 15 times in the case of PAD. Similarly, the range of addresses is reduced by 3 to 12 times on different testbeds. PAD also reduces the hop distance from landmarks by 10–25%.

We also compared our approach with CTP [GFJ+09] against a single landmark in the network. Although CTP is not a point-to-point routing protocol, it is a standard collection protocol that has matured over years. The idea is to see how BVR would have behaved if its multiple trees were based on state-of-the-art CTP and 4BLE: both have been used as a standard for comparison in many recent studies [MSKG10, ALL+09]. CTP and BVR trees are based on similar long-term link estimation concepts. CTP is not designed to provide stable-addressing. In particular, it is optimized for many-to-one scenarios, improves the reliability of transmission and is very aggressive in changing parents after only five unsuccessful transmissions. For PAD, we mostly experienced a similar, and sometimes even bigger improvement in address stability over CTP. Thus, these results validate and are in line with our comparison of PAD and BVR. Hence, we use BVR as main ground for our comparison in this dissertation.



(a) Average Change Rate

(b) Average Magnitude of Change



(c) Average Hop Distance from Landmarks

Figure 5.8 Summary of PAD evaluation: PAD achieves 7 times more stable addressing than BVR under MoteLab's challenging network conditions (notice the logarithmic scale). The error bars represent the *stdev*.

Concluding our comparative evaluation, we have seen that PAD makes significant strides in enhancing the efficiency of tree-construction based virtual addressing in wireless networks. It shows that stable addressing across the network can be achieved without compromising the adaptability of virtual coordinate based routing, which has been the trade of existing routing approaches for a long time.

5.5 Routing on PAD

After discussing the design and experimental evaluation of PAD, we now present a simple routing strategy that can operate on PAD's addresses. While an advanced routing algorithm is not part of our main research contribution in this dissertation, we present a simple design here for completeness. Our goal is not to design an optimized routing protocol but merely to provide an address prediction mechanism and an adaptive routing strategy for the purpose of evaluation to see potential benefits and drawbacks of PAD. On the whole, we introduce some new flavors to make routing over PAD more adaptive and borrow some tactics from the existing approaches as well. Our discussion highlights both.

PAD's design in principle is independent from any specific routing strategy that operates on virtual coordinate based addressing mechanisms. Therefore, depending upon the application requirements, any routing strategy that leverages specific design objectives shall integrate well into PAD. Such strategies could include energy efficient [EV07] and adaptive mechanisms [MSKG10, ALL+09] to maximize routing throughput or multipath [GGSE01] and retransmission [GFJ+09] mechanisms to achieve reliable communications. The approach presented here is a combination of both: It quickly adapts to the underlying link conditions while ensuring reliability by embedding retransmissions and link symmetry tests into the routing decisions.

There are two main elements of our routing approach: (1) We need a mechanism to precisely predict the location of a node in its coordinate distribution, and (2) we need to define a distance function to select the best next hop for forwarding the packet towards a certain destination. While there are well defined distance functions, such as the ones used by BVR [FRZ+05] and LCR [CA06], predicting a node's location in its address distribution is a task we need to deal with. Our choices for both these elements are influenced by our primary design objective, i.e. simplicity. In Section 5.6 we show that even this simple routing strategy over PAD's addresses can reduce hop count, number of transmissions and packet loss in the network.

5.5.1 Address Prediction

In a first step we compose PAD distributions into a meaningful address that can be used to derive routing decisions. For this purpose, we calculate the *mean* over each coordinate distribution in a node's address. Then, sender nodes can use the *mean* address (cf. Section 5.3.3) to forward the packet towards the destination in its virtual coordinate space. In our prototype implementation a node thus advertises these *mean* coordinates in its beacons as a node's address for routing purposes.

As an alternative prediction mechanism, we propose utilizing coordinate *variance* information. The idea behind coordinate variance is to describe a node's location only with respect to those landmarks with stable hop distances over a certain time period. High coordinate variance corresponding to a landmark signifies that a node's hop distance from that landmark varies significantly. Hence, its location with respect to that landmark can be predicted less accurately.

5.5.2 Distance Function

The goal of a distance function is to identify a neighboring node that minimizes the remaining virtual distance to the destination. Other important factors influencing PAD include the availability and link asymmetry (cf. Section 5.5.2.2 and 5.5.2.3 respectively) of a neighbor at the time a packet has to be sent, particularly since PAD does not use long-term link estimation.

5.5.2.1 Minimize Distance

To route packets, we need a distance function that, at each hop, selects the best next hop for the packet to reach its destination. We use a similar mechanism as BVR for

routing except that a data packet now carries the *mean* coordinates (derived from PAD addresses) of the destination instead of BVR's coordinates. At each hop, the destination's *mean* coordinates are compared with the coordinates of all one-hop neighbors. The neighbor, whose coordinates are most similar to the destination, is selected as the next hop for the packet. This process continues until the destination is reached or none of the one-hop neighbors further reduces the remaining distance to the destination, i.e. the current node is closest to the destination in terms of its virtual coordinates. In that case, we use the fallback mode (cf. 5.6.1).

In order to compare coordinates for selecting the best next hop, we propose the combination of the *sum distance* and the *sum of the differences*. The *sum distance* metric signifies the distance of the shortest path (number of hops) from a node S to a destination D via the landmark L. So the best next hop T is the one that minimizes $\bar{d}_k^S(T, D)$, the sum distance between T and D, averaged over a set $\mathcal{C}_k(D)$ of the $k = |\mathcal{C}_k(D)|$ landmarks closest to D [FRZ+05]:

$$\bar{d}_k^S(T, D) = \frac{1}{k} \sum_{L \in \mathcal{C}_k(D)} (\bar{h}(T, L) + \bar{h}(D, L)) \tag{5.1}$$

with the mean distance of neighbor T to landmark L:

$$\bar{h}(T, L) = \frac{1}{\sigma} \sum_{j=1}^{\sigma} h_j(T, L) \tag{5.2}$$

where $h_j(T, L_i)$ is the j-th entry in the history of length σ of hop distances from node T for landmark L. Our implementation includes all landmarks of the network, i.e. $k = \lambda$. We propose a combination of $\bar{d}_k^S(T, D)$ and the *absolute component-wise difference* (cf. Section 2.3.2.1):

$$\bar{d}_k^B(T, D) = \frac{1}{k} \sum_{L \in \mathcal{C}_k(D)} |\bar{h}(T, L) - \bar{h}(D, L)| \tag{5.3}$$

The idea is to choose the smaller one of both distances for routing. Similarly, statistical measures, such as Kullback–Leibler divergence [CBPG11], Hellinger distance [SWWJ08], or total variation distance [DVB01] are possible choices to select a next hop based on its address distribution. However, utilizing these distance measurements for routing over PAD is a research challenge in itself, and a detailed exploration of the design space is beyond the scope of this dissertation.

To compare our prototype implementation of PAD routing with BVR, we only use $\bar{d}_k^B(T, D)$ over the mean coordinates to analyze the true impact of PAD on routing without changing the decision process. In PAD, a neighboring node is only considered a possible next hop if it satisfies the conditions discussed in the following two sections.

5.5.2.2 Link Age

First, we need to determine the reliability of a link with a particular neighbor. Our goal is to derive a minimum reception history that is sufficient to declare a

link reliable for transmission. This is because our design objective is to allow for maximum adaptability to the underlying link conditions in the network. PAD allows us to achieve this objective, while it defines addresses in the form of probabilities and decouples packet forwarding from addressing in the network. As a result, adapting routing decisions to the recent transmission characteristics does not influence the addressing infrastructure.

To check for the availability of a link, we adapt our BRE approach (cf. Section 4.4.1), which determines the fate of the future transmission over a link on the basis of the last three transmissions over the same link. In BRE, a link is declared reliable for transmission if the last three transmissions were successful over that link. This is because in this case the probability of the next transmission over that link being successful is greater than 0.9 [ALL+09]. Although BRE approach is peculiar to unreliable bursty links[11] that cannot be estimated over longer time scales, it is simple and our preliminary evaluation showed that it is certainly effective for long-term stable links as well. Thus, we introduce an *aging* factor for each neighboring link: A link is only considered reliable for transmission if it has an age of 3, i.e., the last three packets were successfully received over that link.

However, these packet forwarding decisions do not change the probability distribution of a node's location. Hence, PAD allows us to maintain a stable coordinate distribution while adapting to link conditions as fast as possible over the duration of three beacon intervals.

5.5.2.3 Link Asymmetry

Link asymmetry is a major issue in wireless networks where routing demands each packet to be acknowledged by the receiving neighbor. To exclude neighbors with asymmetric links from the routing process, the prevalent approach is to calculate the bidirectional link quality based on beacons and transmission statistics. The resulting routing path is the one that offers the minimum number of transmissions for the data packets and the corresponding acknowledgments.

In PAD, each node S maintains a set of those neighbors to which it has symmetric links. A neighbor T is considered to be alive and on a symmetric link as long as at least one of its beacons arrives within σ beacon intervals and lists S as a neighbor of T. Consequently, a neighbor's validity expires automatically after σ beacon intervals, if no beacon is received from it during this time. Another mechanism to test for link symmetry is to actively monitor ACKs over a link. ACKs are a useful and automatic test for link symmetry and are also employed in our prototype implementation.

5.6 Routing Results

We compare our simple routing strategy over PAD with S4 and BVR. We use the *number of transmissions* as our metric for comparison because it is the prevalent routing metric in energy constrained sensornets [MSKG10, ALL+09, GFJ+09].

[11] The links that show dynamics in their reception rates at a subsecond granularity.

5.6.1 Experimental Setup

Our experimental setup is similar to the one for our PAD evaluation in Section 5.4 except that now a randomly selected set of sender-receiver pairs is defined. Each sender sends 500 packets to its destination, one sender at a time. All the data traffic happens after an initialization period of 15 minutes. The sender nodes are only informed once about the destination's virtual address at the beginning of the packet burst. Therefore, if the address of the destination changed during transmission, the sender node would still route packets on the outdated address. Moreover, the sender and receiver nodes might be landmarks themselves as well. We allow five routing level retransmissions[12] for each possible next hop, both in BVR and PAD.

Our prototype implementation of routing on PAD addresses shares another aspect with BVR, i.e. the fallback mode and scoped flooding (see [FRZ+05] for details). The idea of the fallback mode is that in case a packet reaches a dead end, it is forwarded to the landmark closest to the destination. In case of PAD, as it does not maintain any explicit parents, routing towards the closest landmark is performed by selecting a neighbor that offers the minimum hop distance towards the landmark closest to the destination and qualifies the prerequisites such as link age and symmetry discussed in Section 5.5.2. Each node receiving the packet on the way to the landmark tries the normal greedy routing mode again and continues in fallback mode in case this fails again. If the packet reaches the landmark, this initiates a flooding of the packet with the scope as high as the path length (*hop distance*) from the landmark to the destination node (revealed by the destination's address). This mechanism incorporates the hope that the destination will receive the packet until the flooding scope has been reached. The inclusion of scoped flooding in PAD is not a way to enhance performance but to provide a backup path and to complete the implementation for a fair comparison with BVR.

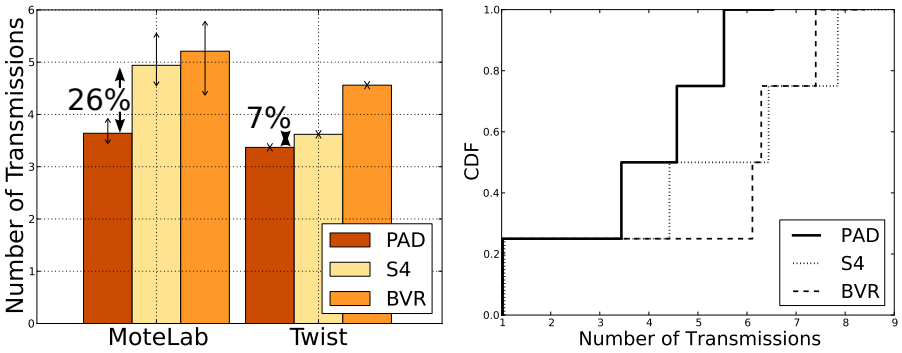
5.6.2 Number of Transmissions

As our prototype implementation is for sensornets, our key performance metric is the *number of transmissions* required by a packet to reach its destination. Other factors like throughput are not considered here.

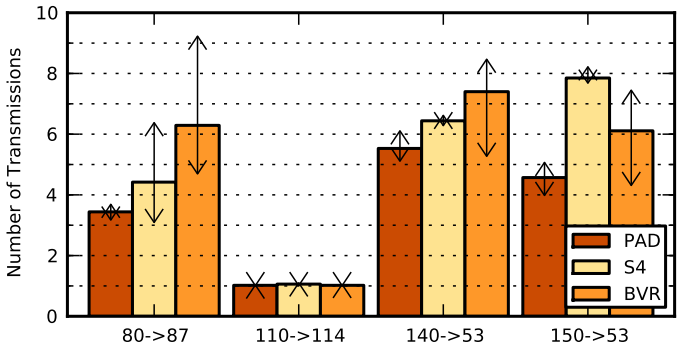
Figures 5.9(a) summarizes our results across the three testbeds. To observe the stability of results over time on MoteLab, we repeated our experiments 5 times for each protocol. The bars in Figure 5.9(a) show the average of 5 experiments while the error-bars show the highest and the lowest results among these experiments. The results clearly indicate that on MoteLab PAD outperforms both S4 and BVR by decreasing the number of transmissions by at least 26%. However, due to very stable link conditions on TWIST, the margin of improvement is just 7%. Figure 5.9(b) shows the CDF for the *number of transmissions* and Figure 5.9(c) details the results for a subset of sender-receiver pairs on MoteLab.

To understand the sanity of these results, we need to revisit a few mechanisms of S4. First, S4 is very conservative in its structure and does not rapidly adapt its topology to the changing underlying conditions in the network. Therefore, it

[12]This is the default retransmission count in the original implementation of BVR.



(a) Average Number of Transmissions (b) CDF of the Number of Transmissions in MoteLab



(c) Number of Transmissions for a subset of sender-receiver pairs in MoteLab

Figure 5.9 A simple routing strategy over PAD reduces the number of transmissions in the network when compared with BVR and S4. The bars represent the average of 5 experiments and the error-bars show the highest and the lowest results.

employs retransmission of beacon packets to sustain its hybrid topological structure and maintain a small routing stretch. Second, S4 uses a link quality threshold of $PRR = 30\%$ (calculated using a passive WMEWMA estimator) to accept a link into its routing process. Using such links in a network dominated by low-quality links, without assessing their quality in the short-term, understandably decreases the number of routing choices and increases the number of transmissions in the network. In contrast, PAD incorporates rapidly changing conditions in its fuzzy addresses and assesses links based on very recent transmission conditions. Both these mechanisms of S4 explain the diversity of the results across different testbeds (cf. Figure 5.3). For example, the margin of improvement is quite high on MoteLab, whereas on TWIST the results are very comparable for all the three protocols.

We also evaluated the impact of landmark failures on transmission characteristics of PAD, such as the fraction of routes that directly arrived at the destination compared to the fraction of routes that required scoped flooding. Our results show similar trends as BVR's results [FRZ+05].

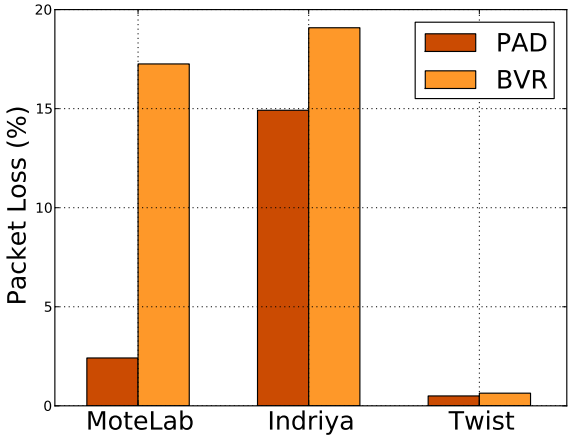


Figure 5.10 Delivery reliability: PAD reduces the packet loss on each testbed due to a high adaptivity and a smaller magnitude of change in its addresses.

5.6.3 Reliability

Figure 5.10 shows the results of the packet loss comparison. On each testbed, PAD reduces packet loss in the network. These packets never reach their destination even after scoped flooding and are finally dropped. We observed that the majority of these losses are not due to link failures. Please note that the number of retransmission attempts at the routing level is equal for PAD and BVR. Moreover, these packets are not dropped due to contention, because we only enable one sender at a time.

We gave a detailed account of this behavior in Section 5.4.3.2, where we show that address changes in the case of PAD usually result in a minimum shift in the virtual space. This is because the magnitude of changes in PAD addresses is significantly smaller than in BVR. As a result, in PAD it is more probable that a packet would reach the vicinity of its destination even if it was routed according to an outdated address. Whereas, in the case of BVR, changes in the addresses could result in a major relocation of a node in the virtual address space. Hence, if routed towards an outdated address, it is unlikely for a packet to reach its destination. This is true even for the scoped flooding mechanism, if the address changes affect a node's distance from its nearest landmark. For example, if a node moved from 2 to 4 hops away from its nearest landmark, the scoped flooding mechanism would fail to deliver a packet to the destination, since the packet would be dropped after having traveled two hops from the landmark. On TWIST, the packet loss is minimal because of the very good connectivity and the very low average number of transmissions for each packet, i.e. 2.

Concluding our routing results, we also evaluated the impact of landmark failures on transmission characteristics of PAD, such as the fraction of routes that directly arrived at the destination compared the fraction of routes that required scoped flooding. However, our results show similar trends as BVR's results [FRZ+05].

5.6.4 Memory and Communication Overhead

Here we take a closer look at PAD's memory and communication overhead. Against the baseline of BVR, PAD introduces larger node addresses as they contain a node's

coordinate history encoded as a probability distribution. The size of PAD addresses heavily depends on the number of landmarks λ and the history size σ (see Section 5.3.1). For example, 5 landmarks and 5 smallest paths in the history of the last σ coordinates result in an address length of 25 bytes.

These larger node addresses in PAD impact the following three communication scenarios:

- **Local beacon updates:** In this case, PAD allows to trade off transmission overhead against memory overhead. (1) Either a node's PAD address is included in its beacons, which increases the transmission overhead. (2) Or only the node's most recent coordinates are transmitted such that the neighbors that receive the beacons can compute the node's PAD address themselves (c.f. Section 5.3.3). This increases the CPU and memory overhead but does not introduce any transmission overhead against the baseline of BVR's beacon header. Moreover, one has to consider that PAD saves the transmission overhead of all the additional bytes appended with each data and beacon packet by BVR's link estimator.

- **Global address update:** This update is required in the network's address database whenever a node changes its PAD address. The database interaction is beyond the scope of discussion in this dissertation. However, to put this overhead estimation into perspective, one has to consider that PAD needs significantly less address updates.

- **Data transmissions:** Finally, each data packet needs to carry the destination address in its header. In our current implementation, we are only using the *mean* for each coordinate distribution in a PAD address. Hence, in its current state, PAD does not introduce additional overhead against the baseline of BVR's data-packet headers.

Similarly, the following five parameters are the major contributors to the beacon size and memory consumption of PAD.

- **History size (σ):** The history of coordinates maintained by PAD to compute its address.

- **Distribution size (δ):** The range of hop distances from each landmark.

- **Neighbor table size (ν):** The maximum number of neighbors maintained by a node.

- **Trace length (τ):** The number of node IDs in the landmark-to-node traces contained in beacons for loop avoidance.

- **Number of landmarks (λ):** The number of landmarks in the network, i.e. the dimensionality of the virtual coordinate system.

Parameter	Memory Overhead (bytes)	Transmission Overhead (bytes)
History size (σ)	λ	-
Distribution size (δ)	2	2
Table size (ν)	$2 + \lambda + \sum_{i=1}^{\lambda} \delta_i$	-
Trace length (τ)	-	λ
Landmarks (λ)	$1 + \delta + \sigma$	τ

Table 5.2 Memory and transmission overhead estimation: The table shows how to calculate memory and transmission overhead caused by increasing the corresponding parameter by 1.

The **history size** refers to the history of current coordinates used to calculate PAD's address. Therefore, by increasing the history size from σ to $\sigma + 1$, the memory consumption is increased by λ bytes, as each new address vector requires as many bytes as there are landmarks in the network. However, history size does not add to the transmission overhead. In contrast, BVR uses expensive link estimation and does not maintain any history of coordinates.

The **distribution size (δ)** refers to the number of hop distances from a landmark and their probabilities in PAD. It takes 2 bytes to store each hop count from a landmark and its relative frequency in the distribution. However, there is one such distribution structure for each landmark in the network. Hence, a change from δ to $\delta + 1$ for a landmark leads to an increase in memory of 2 bytes but does not have any impact on the size of the beacons. However, the packet size to update the address in the address-database would increase by 2 bytes. Compared to that, BVR has an address of constant length, i.e. λ.

A change in the **routing table size (ν)** has a major effect on the memory footprint. Our neighbor table stores the following information regarding each neighbor: (1) node ID (2 bytes) (2) current coordinates (λ bytes), and (3) PAD address ($\sum_{i=1}^{\lambda} \delta_i$ bytes). In contrast, BVR additionally stores path quality information for each neighbor and also maintains a separate link estimator table at least as big as the routing table.

The **trace length (τ)** increases the size of the beacons. By changing the trace length from τ to $\tau + 1$, the packet size increases by λ bytes. Finally, the **Number of landmarks (λ)** affects the size of PAD's addresses by δ bytes and of the current coordinates by 1 byte. Therefore, the size of the history data structure increases by σ bytes as well when changing λ to $\lambda + 1$. Similarly, each landmark requires a trace in the beacon, and hence, increases the size of each beacon by τ bytes. Table 5.6.4 summarizes the formulas to calculate the memory and transmission overhead for the different parameters.

In general, PAD allows to trade off transmission overhead against memory overhead by choosing how address information is disseminated in beacons. This trade-off may need to be evaluated depending on concrete deployment and application scenarios of PAD.

5.7 Discussion and Related Work

The need for location independent addressing and routing schemes has long been realized since the emergence of multihop wireless communication systems such as ad hoc, mesh and sensor networks. These schemes are known for their simplicity, self-configurability, scalability, and for maintaining a constant routing state on each node in the order of the one-hop neighborhood size, making them particularly appropriate for resource-constrained sensornets. PAD has three complementing features that distinguish it from conventional location free addressing and routing approaches:

- It assigns fuzzy but adaptive addresses to nodes instead of sharp coordinates by analyzing link variability patterns without link estimation and explicit trees in the network.
- It decouples addressing from routing allowing for quick adaptation of routing algorithms based on recent network conditions without compromising the stability of addressing.
- It embeds the information about all possible paths leading to a node in its address.

For our prototype evaluation we used BVR's greedy routing mechanism. However, we believe that the same ideas of probabilistic addresses can be used with S4's inter-cluster routing approach as well. The functioning and performance of S4 is strongly dependent on a stable topology in which nodes can accurately estimate their distance from the nearest landmarks. To achieve this high level of stability and resilience S4 employs costly mechanisms, such as Resilient Beacon Distance Vector (RBDV), which retransmits a broadcast beacon until a specified number of neighbors have forwarded the same beacon. As a result, as we observed in Section 5.6, S4 can accomplish its goal – achieving a small routing stretch – without excessively increasing the *number of transmissions* only in very stable network conditions (e.g. TWIST). However, in testing conditions (e.g. MoteLab), S4 has to pay a high price of increased *number of transmissions* in the network for maintaining smaller routing stretches. PAD tolerates the need to maintain such a stable and resilient topology by assigning fuzzy locations to nodes and by allowing to adapt routing to very recent link conditions.

In the following we discuss the major related efforts in sensornets and meshnets.

5.7.1 Sensornets

LCR [CA06] and BVR [FRZ+05] are two very similar and notable implementations of virtual coordinate based addressing in sensornets. Both LCR and BVR provide extensions based on link estimation for stable addressing in the presence of unreliable links in wireless networks. However, in Section 5.4.3.1 we already observed that long-term link estimation suffers in networks with challenging conditions such as experienced in MoteLab, as BVR's addressing showed instability and requires

Aspect	BVR	S4	PAD
Link estimation	overhearing	overhearing	none
Addressing	sharp	sharp	fuzzy
Node location	virtual coordinates	nearest landmark	probabilistic address
Routing	greedy	cluster-based	greedy

Table 5.3 Comparative Overview: Protocol aspects of BVR, S4 and PAD.

frequent address updates throughout the network. Table 5.7.1 briefly compares different protocol aspects of PAD, BVR and S4.

GEM [JS03] introduces a graph-based scalable addressing scheme. However, it employs a complex recovery process, in which a potentially large number of nodes in the system must recompute their addresses in case of node failure or radio link deterioration. In contrast, PAD provides an elegant solution to maintain address stability even in lossy networks.

5.7.2 Meshnets

NoGeo [RRP+03] and DART [EFK07] are location-independent addressing schemes for **meshnets and MANETs**. In NoGeo, nodes determine their coordinates in the Cartesian coordinate space through an iterative relaxation procedure with reference to a set of parameter nodes. Its initialization scheme requires to maintain a node state in the order of $O(n)$ on $O(\sqrt{n})$ nodes, which is not feasible in sensornets. DART establishes address trees where leaves of the address tree represent actual node addresses, while each inner node represents an address subtree. However, this approach heavily emphasizes the maintenance of the address trees and is evaluated in high-level simulations. It is not yet clear how practical this approach is with regard to the rate and magnitude of change in coordinates observed in real deployments.

Overall, PAD provides a flexible addressing platform that can host different routing strategies depending on application requirements while maintaining the scalability advantages of tree-based routing infrastructures.

5.8 Summary

We presented a robust and scalable addressing mechanism for wireless networks. When compared with other addressing mechanisms, PAD increases the stability and reduces the magnitude of change in addresses at the low cost of larger beacon packets. An adaptive routing strategy over PAD allows quick adaptation of the routing paths based on very recent link conditions. Our results from testbed environments demonstrate that even an unoptimized version of routing over PAD can enhance packet delivery over multiple hops. Similarly, our tests under challenging environments such as in MoteLab show that PAD can realize its advantages in real world deployments.

In general, PAD provides a number of design choices to trade off transmission overhead against memory overhead by choosing how address information is disseminated

in beacons. The first option is to include a node's PAD address in its beacons which increases the beacon size. The second option is to only transmit a node's current coordinates instead of the aggregated PAD address. In this case, the neighbors that receive the beacon need to store a history of theses coordinates and compute the PAD address themselves (c.f. Section 5.3.3), which increases the CPU and memory overhead.

PAD is particularly suited for challenging network conditions. By assigning fuzzy addresses to nodes instead of sharp coordinates and by employing an adaptive routing strategy, PAD adapts its routing paths to the most recent link conditions in the network. As a result, it outperforms the stat-of-the-are in point to point routing in sensornets which overwhelmingly rely on the presence of links with stable quality. The design of PAD is not dependent upon the presence of stable links, which is often an invalid assumption due to the notoriously frequent variations of wireless links. PAD's superior performance versus BVR and S4 on three testbeds with varying link conditions proves its utility as a robust addressing scheme in multihop wireless networks.

6

Exploring General Applicability

So far, we evaluated the design of the proposed approaches in sensornets, i.e., IEEE 802.15.4 based link layer service. These approaches leverage link characteristics, such as short-term dynamics and burstiness, which are similar across different link layer technologies [SKAL08, ABB+04, AWK+11b]. Similarly, it is evident that all the three approaches do not rely on a specific link layer service or technology. This is because our algorithms and metrics are based on network level measurements that only extract information from the layer-3 headers.

Although our designs are independent of link layer properties, we still cannot make any assumptions regarding the feasibility and performance of these approaches on a different link layer technology. Rather, we need to demonstrate this empirically for two main reasons: (1) IEEE 802.15.4 is very different from other link layer technologies such as IEEE 802.11: For example, it operates on a constant bit rate, supports very low data rates (few tens of Kbps compared to tens of Mbps in IEEE 802.11), and its design is optimized for energy efficient operation of power constrained devices. (2) We are not yet clear if and how these aforementioned differences in link layer technologies can impact the proposed approaches. In this chapter, we thus aim at evaluating the proposed approaches in IEEE 802.11 based wireless networks to show that this dissertation owns a broader relevance in the wireless networking domain.

The underlying observation that forms the basis of our work in this chapter is that multihop wireless networks, such as sensornets, MANETs and meshnets, although different, share some common characteristics. All these networks exhibit link dynamics. Protocols designed for these wireless networks must overcome the challenge of link dynamics and the resulting churn in network topology. Due to structural and topological similarities, protocols developed for one class of wireless network should also be applicable in the other classes. However, network-layer protocols are usually developed for and tested in only one class of wireless network due to the lack of a platform that allows testing of protocols across different classes of networks. As a result, we unnecessarily constrain the range of settings and scenarios in which we test network protocols.

The chapter makes the following two main contributions:

- In order to avoid tedious re-implementation effort associated with testing protocols in a different wireless network class, we present TinyWifi, a platform for executing native sensornet protocols on Linux-driven wireless devices. TinyWifi builds on nesC code base that abstracts from TinyOS and enables the execution of nesC-based protocols in Linux. Using this abstraction, we expand the applicability and means of protocol execution from one class of wireless network to another without re-implementation. The bulk of this chapter discusses the design and evaluation of TinyWifi because it enables the execution network protocols across multiple PHY-Link layers and thus forms the basis of our claims regarding the general applicability of our mechanisms.
- We evaluate PAD in meshnets and MANETs. We only evaluate PAD because (1) it combines all the main concepts presented in this dissertation, and (2) it enables point-to-point routing which is the prevalent communication paradigm in meshnets and MANETs. Our evaluation makes use of both simulations and testbeds. For our experimental evaluation, we use TinyWifi to compare PAD's performance on IEEE 802.15.4 and IEEE 802.11 based testbeds using a single nesC implementation. On the other hand, our simulations are focused on evaluating PAD in mobility scenarios, a key characteristic of MANETs.

The rest of this chapter is structured as follows: Section 6.1 details the design, implementation and evaluation of TinyWifi. Then, we evaluate PAD is Section 6.2. Finally, we summarize the discussion in Section 6.3.

6.1 TinyWifi

Sensor-, ad hoc-, and mesh-nets, represent vastly different classes of wireless networks. They not only use different types of radios and link layers but also OS, hardware platform, programming/runtime environment, and application scenarios. While different, they also share some commonalities:

- Dynamic and bursty links due to radio interference and other physical influences,
- Use of multihop protocols to reach nodes not within radio range,
- The intended use cases demand a reliable and scalable communication infrastructure, and
- They are self-organizing in arbitrary and temporary network topologies.

These similarities lead to an important question: How well can the algorithmic concepts, proven methods, and protocols from one class of wireless network be adapted to the other classes of wireless networks? In general, research efforts, such as on link estimation [FRZ+05, FGJL07], routing [MSKG10, MWQ+10] and

addressing [FRZ+05, AVL+11], explore the feasibility of these protocols in one class of networks and implicitly assume their applicability in the other, based on the above mentioned similarities. This assumption is rarely validated due to the lack of a common development platform that allows us to test the protocols across the vastly different classes of wireless networks.

To understand the performance of the protocols and their applicability across multiple wireless network classes, a common programming environment and a runtime platform is essential. It is well understood that incompatible application requirements and unequal resource constraints make for a significant diversity among these different classes of networks. However, this diversity, in most cases, only demands appropriate adaptations in operational parameters of the underlying protocols while the core mechanisms still remain the same [FGJL07, WTC03, ALL+09]. For example, to account for the underlying resource availability in different networks, routing protocol configurations may only need to adjust parameters such as routing table sizes and the frequency of routing updates. Nonetheless, the metrics used to select a next hop and establish routing paths - the core and the most complex mechanisms of a routing protocol mechanism - remain the same: ETX (expected transmission count) [DCABM05] is the most prevalent routing and link metric both in sensornets and meshnets [WTC03]. Moreover, a common development platform will help determining the impact of lower layer technologies, such as medium access, coding, and modulation schemes, which are different in IEEE 802.11 and 802.15.4 standards, on the performance of these core protocol mechanisms.

As a first step towards such a platform, we introduce TinyWifi, a TinyOS platform supporting Linux driven devices and thereby the IEEE 802.11 based Wi-Fi standard. The utility of TinyWifi is twofold:

- It is a runtime platform that allows direct execution of protocol libraries in three different network classes.
- It makes the very rich and mature protocol repository of TinyOS available for a broader scope of wireless research.

TinyWifi supports a wide variety of Linux kernel derivatives representing all major Linux distributions such as OpenWRT, Debian, Slackware and Ubuntu. We (and a few other research groups) are using TinyWifi[1] to run nesC protocols in meshnets.

We evaluate the correctness of our TinyWifi implementation by comparing two different implementations of the Collection Tree Protocol (CTP) [GFJ+09], one in nesC and the other in Click [KMC+00]. Our comparison proves the equality of these two implementations and demonstrates the utility of TinyWifi as a customary wireless research and runtime platform. During our evaluation on an IEEE 802.11 based testbed, we observed that TinyWifi is particulary useful for (1) evaluating prototypes, (2) fine-tuning protocol parameters and (3) establishing multiple performance metrics in different classes of wireless networks without re-implementation.

[1]The source code of TinyWifi is available for download at http://www.comsys.rwth-aachen.de/research/projects/tinywifi/.

6.1.1 Preliminaries

We first provide necessary background by briefly introducing TinyOS. Then, we present the overall design of TinyWifi. Finally, we highlight the key features of our TinyWifi implementation.

6.1.1.1 TinyOS

TinyOS is the de facto standard operating system for sensornets. It has an event driven architecture which enables development of energy-efficient sensornet applications. It has been in active research and development over the past decade and its novel protocol mechanisms, such as in link estimation, routing and addressing, are developed and actively used worldwide.

Programming Model

The Programming model of TinyOS is based on a component based nesC language. nesC is an extension of C-language which supports the TinyOS concurrency model and allows to build components and link them together in an application configuration. The TinyOS system, libraries, and application are all developed in nesC. Following are some of the important programming constructs of the nesC Language.

Interfaces are interaction points between components. Each component is a collection of *commands* and *events*. *Commands* are the services offered by the components. *Commands* of an interface are implemented by the component(s) providing that interface. *Events* signal the completion of services. *Events* of an interface are implemented by the component using that interface.

Components encapsulate a specific set of services specified by interfaces. There are two types of components in nesC, *modules* and *configurations*. *Modules* offer services by implementing interfaces. Every *module* in nesC has a *specification* and *implementation*. *Specification* lists the interfaces used and provided by the module. *Implementation* implements the commands and events of the provided and used interfaces, respectively. *Configuration* wires the modules together in an application by connecting the interfaces used by the modules with their respective implementations provided by other modules. An application in TinyOS is a set of modules and a wiring specification that connects these modules with each other.

Tasks are functions whose execution is deferred until no event handlers are running. Tasks do not preempt each other. Once a task is scheduled, it will always run to completion. Commands and events post tasks for immediate return and to defer lengthy processing. The Scheduler in TinyOS schedules the execution of tasks in FIFO order. It also maintains a finite task-queue with every task having its own reserved slot in the queue. A task can only be reposted once the previous post of the task has been dispatched for execution by the Scheduler. Tasks can be preempted by hardware event handlers.

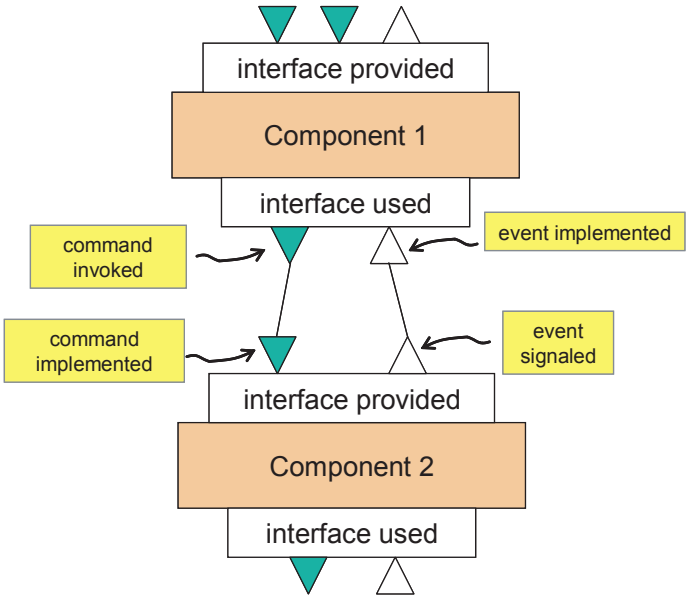


Figure 6.1 TinyOS programming model: Each component provides and uses interface(s). A component that provides an interface must implement all its commands. A component that uses interface must implement all its events. Commands and events are decoupled from each other resulting in the split-phase operation of TinyOS.

There are two types of events in TinyOS i.e. *synchronous* and *asynchronous*. *Asynchronous* events usually represent hardware interrupts that can preempt other events and tasks. *Synchronous* events, on the other hand, are used to complete the *split-phase* operation of TinyOS. Split-phase execution decouples the commands and events from each other, thus, there are no blocking calls in the system. In the first phase of the split-phase operation, a service-user component calls for a service by calling a command implemented by the service-provider component. The service-provider component posts task(s) to perform the ordered service and returns immediately. Once the execution of a service is complete, the service provider component signals an event to the user component indicating the completion of service and hence the second phase of the split-phase operation.

Architecture

TinyOS does not perform traditional operating system functions like process management, memory management, and virtual memory management. Applications and the operating system use a shared stack. Hence, TinyOS is not an operating system in a tractional sense, it constitutes a set of open source system components that assists developers in the development process and provide easy access to the underlying hardware resources. Application developers can easily and independently modify the TinyOS system components depending upon the needs of their applications.

TinyOS supports multiple sensor node hardware platforms. A platform in TinyOS is a set of chips and glue code that holds these chips together. For example, Mica2 [Tec07] platform is made up of an Atmega128L micro-controller and a CC1000 radio-chip.

The architecture of TinyOS is divided in three abstraction layers [HPH+05]: Hardware Presentation Layer (HPL), Hardware Abstraction Layer (HAL), and Hardware Independent Layer (HIL). The modules at HPL are hardware-dependent and present the capabilities of the underlying hardware while hiding its intricacies. In contrast, the modules at HAL and HIL are platform independent and can be used across different hardware platforms. Protocols and applications are built on top of HAL and HIL. TinyOS can be extended to new hardware platforms by providing the corresponding HPL support for that platform.

TinyOS owns a very rich protocol repository for IEEE 802.15.4 based networks. Among the most prominent protocols developed for TinyOS are CTP [GFJ+09], 4BLE [FGJL07], BVR [FRZ+05], S4 [MWQ+10], BCP [MSKG10], and PAD [AVL+11]. Supporting flexible networking structures and achieving reliable and energy-efficient multihop communications drives the design philosophy of these protocols.

6.1.1.2 Design Overview

TinyWifi enables the execution of protocols developed in nesC (for TinyOS) on Linux based Wi-Fi devices, i.e., nodes in meshnets. The key idea is to exploit the modularity of the TinyOS hardware abstraction architecture: TinyWifi replaces the existing TinyOS core at HPL to provide the exact same hardware independent functionality and interfaces as a regular sensor node platform (cf. Figure 6.2). For example, the active messaging interface for IEEE 802.15.4 based CC2420 chips is replaced with a socket based communication interface for Linux networking. Similarly, hardware timers are replaced with Linux timers.

This seamless integration enables TinyWifi to export the resources of typical Linux network devices such as large memory, more processing power, and higher communication bandwidth to the sensornet protocols developed in nesC. However, this transition from mote-class devices to Linux-driven nodes at HPL is not straight forward [Kir10, AKL+10]. Apart from handling hugely different link layers, TinyWifi has to deal with completely different hardware platforms, programming and runtime environments, and computational resources as discussed in Section 6.1.2.

TinyWifi runs as a Linux user space process. It is easy to use and provides simple command-line primitives such as *make linux* and *make linux run* for compiling and executing protocols. The TinyWifi specific code integrates seamlessly into the existing TinyOS source tree. Using TinyWifi as a development platform, any protocol that is written in C or nesC language can be executed both in IEEE 802.11 and 802.15.4 based networks.

6.1.1.3 Key Features

TinyWifi is centered around four design features:

- **Transparency:** Existing sensornet protocols and algorithms developed in nesC shall not break when we run them on Linux based platforms despite the change in the underlying platform characteristics, such as medium access technologies and hardware capabilities.

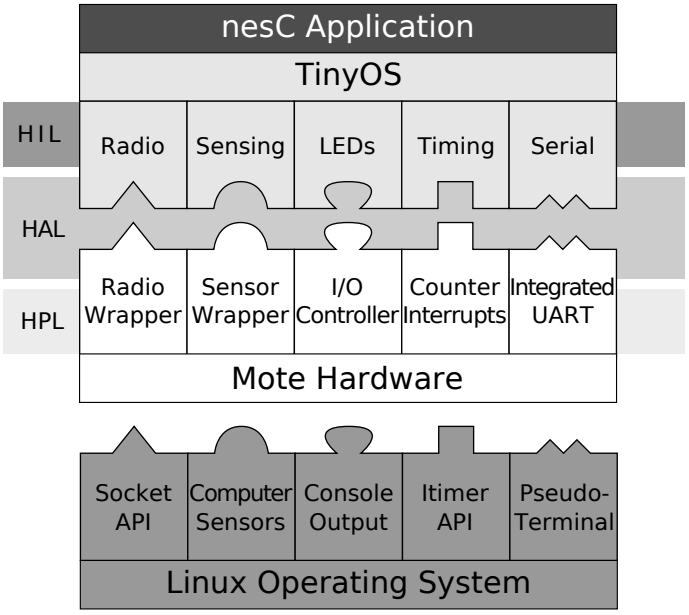


Figure 6.2 TinyWifi Architecture. The hardware abstraction layer (HAL) translates hardware independent functionality (HIL) to the device specific modules of the hardware presentation layer (HPL). TinyWifi replaces the hardware dependant modules at the HPL layer with its corresponding Linux based implementation of HPL components.

- **Versatility:** The implementation should be adaptable to the characteristics of the target platform, for example, whether to encapsulate TinyOS packets in UDP datagrams or bypass the network stack and send them directly over the wireless interface.
- **Usability:** No modifications should be necessary for nesC protocols and the target platform (i.e., Linux) to function. In other words, TinyWifi should be directly deployable in any network that supports Linux based nodes.
- **Adaptability:** TinyWifi should expose the additional capabilities, such as larger memory and processing power, of the Linux platform to the TinyOS protocols.

6.1.2 Detailed Architecture

We now describe the detailed architecture of each component in TinyWifi.

6.1.2.1 Radio Communication

Radio communication is the most vital service and the pivotal difference between sensornets and meshnets at the MAC and PHY layers. TinyOS provides an *active messaging* service [HSW+00] on top of a mote's low-power radio chip such as CC1000, CC2420 etc. An active message contains the identification number of the user-level handler, and the data payload is passed as arguments. The network is modeled as a pipeline and there are no additional buffers used to store messages. Therefore, the

handler is responsible for accepting the message from the network and processing it quickly to be able to receive the next message. TinyWifi replaces this active messaging layer with its own communication service. It provides two flavors of communication services on top of the IEEE 802.11 based network interface: UDP based overlay and direct MAC access.

In UDP based overlay communication, we encapsulate TinyOS messages in UDP packets using datagram sockets. We broadcast UDP packets but suppress routing by adjusting the TTL-field so that packets are only received by TinyWifi nodes within the radio range. This flavor of communication has four key advantages:

- It is simple to implement and very useful for initial debugging and testing.
- It maximizes portability.
- It minimizes interference with different applications on the network.
- It allows direct execution of TinyWifi without negotiating special kernel level privileges.

However, UDP based communication has two main disadvantages:

- It introduces significant processing overhead in processing each packet at the IP and UDP layers which is irrelevant for TinyWifi.
- It does not provide direct access to the wireless interface to utilize important information, such as RSSI and LQI, which might sometimes be essential for higher layer protocols.

For this reason, we provide an interface that utilizes *raw sockets* to enable direct access to the underlying wireless interface. In the current TinyWifi implementation, this interface is the default communication device.

6.1.2.2 Split-Phase Operation

TinyOS employs split-phase operations [GLC05] for system calls, which is a significant departure from how Linux handles its system calls. The key idea of a split-phase operation is to account for the mote's concurrency and avoid blocking-calls in the system. Many system services, such as sending/receiving a packet, are completed in two phases. A command that starts a system service returns immediately while the completion of that service is signaled later via a *callback* event (cf. Section 6.1.1.1). This mode of operation allows TinyOS to process multiple services and the main program in parallel using concurrent processing hardware.

TinyWifi supports both blocking system calls and split-phase operations. The support for blocking system-calls in Linux is trivial (i.e., it is built on native blocking calls). However, to mimic the split-phase programming and runtime operation of TinyOS, we use threads to monitor I/O related operations that run in parallel with the CPU, for example on network cards. When an application module needs to

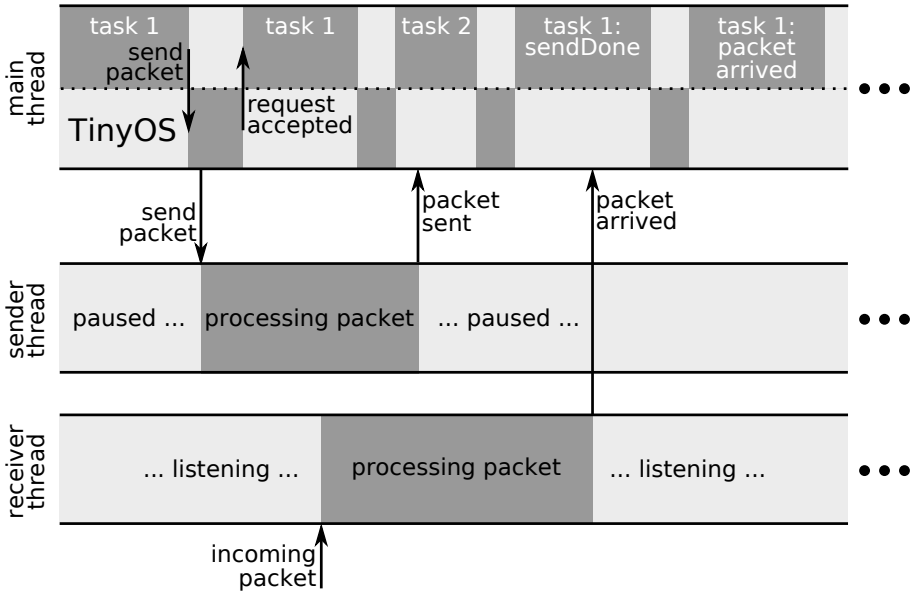


Figure 6.3 Split-phase operation: Using two parallel threads, e.g. a sender and a receiver in the case of radio communication, we achieve the split-phase functionality of TinyOS in TinyWifi.

perform an I/O operation, the corresponding thread is activated and the application continues with its own execution. The completion of these parallel processing threads is then indicated via a Linux signal, which in turn triggers the main TinyOS thread.

Figure 6.3 shows the split-phase operation of TinyWifi for radio *send* and *receive* primitives. A sender and a receiver thread are responsible to handle the respective requests from applications and later signal their completion. The provision of both, the blocking system calls and the split-phase operations, in TinyWifi allows developers to choose a mechanism appropriate for their protocols and applications.

6.1.2.3 Timers

The accuracy of timer operation is critical for the functioning of protocols and time synchronization mechanisms. On the sensor-motes, protocols can directly access hardware counters and timers but this is generally not done by the protocols on Linux based network devices.

The TinyOS timing functionality is based on the hardware timers present in microcontrollers. A sensor-node platform provides multiple realtime hardware timers to specific TinyOS components at the HAL layer - such as alarms, counters, and virtualization. Once configured, these timers trigger an interrupt in the future without the need for continuous monitoring.

Although our target devices provide hardware timers as well, user space applications have no access to them. Therefore, we use Linux's *itimer* library. This library only provides a single realtime timer to each process running on a Linux kernel. However, TinyWifi requires multiple timers to cater the needs of multiple protocols running inside one TinyWifi process, such as link estimators and routing. Therefore, we

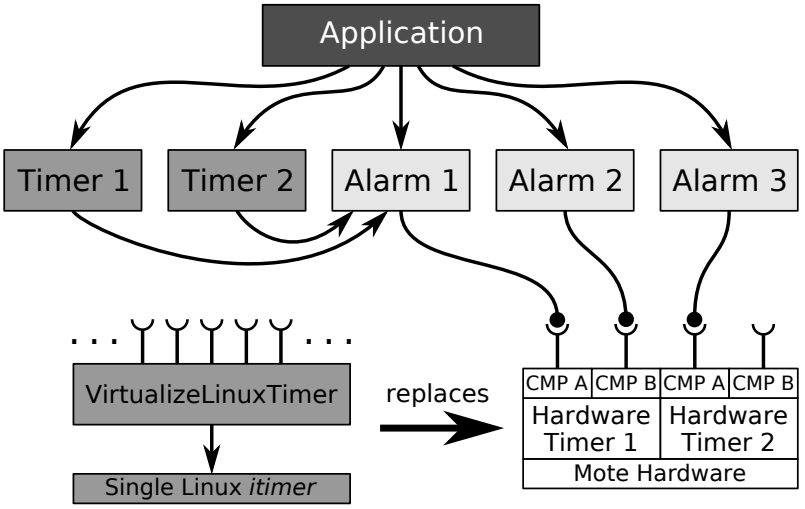


Figure 6.4 Timers: The TinyWifi timer implementation provides several instances of alarms and timers because Linux only provides a single realtime timer per process.

introduce a new *VirtualizeLinuxTimer* component that virtualizes a single *itimer*. This component provides multiple instances of the new *LinuxTimer* module. Figure 6.4 shows the concept of virtual timers and alarms on top of a single *itimer* that replaces the hardware timers of a mote. This virtualization of a single timer is achieved by maintaining a delta-queue, sorted in the order of time, of the registered timer events. The *itimer* is then rescheduled to the most significant event in the queue, i.e., the event at the front end of the queue. This way, we provide timing functionality analogous to typical mote platforms.

6.1.2.4 Miscellaneous Services

Radio communication, split-phase operation and timers make up the major pieces of our design. However, there are certain functionalities, such as serial communication and debugging support, peculiar to motes that are used by the majority of sensornet applications. TinyWifi also provides these functionalities to (i) enhance *usability* by enabling full fledged TinyOS support in meshnets, and (ii) to ensure a *transparent* application transition between TinyWifi and TinyOS.

Serial Communication

The majority of TinyOS applications uses the serial communication for mote-to-PC data exchange. In order to provide a similar functionality, i.e., serial active messaging on a TinyWifi device, TinyWifi uses a Linux pseudo terminal. As with typical motes, an unaltered serial forwarder based on the C programming language connected to the pseudo terminal allows for sending and receiving serial data to and from a TinyWifi node.

Testbed	Available Nodes	Node Degree	Radio	Path Stretch
UMIC-Mesh	35	4	802.11	$\backsim 3$
Indriya	127	18	802.15.4	$\backsim 3$

Table 6.1 Testbed Characteristics: UMIC-Mesh is an IEEE 802.11 based meshnet while Indriya is a TinyOS based sensornet. *Node Degree* refers to the average number of one-hop neighbors. *Path Stretch* refers to the average number of hops between two non neighboring nodes, derived from the connectivity graphs.

Sensing and Debugging

Since our focus is the testing of network protocols, sensing is a subordinate issue. Nevertheless, we do supply dummy sensor implementations to allow for TinyWifi to be used out of the box.

In addition to the *printf* library to output debugging information through the serial interface to an attached PC and displayed in a human readable manner, TinyOS provides *dbg* functions to print additional information. In the TinyWifi implementation, we print those messages directly to the standard output. Similarly, to indicate the status of a physical mote to a developer, motes are equipped with LEDs. TinyWifi provides pseudo-LEDs: Messages are sent to standard output similar to the debugging mechanism of the TOSSIM [LLWC03] simulator.

6.1.3 Evaluating TinyWifi Implementation

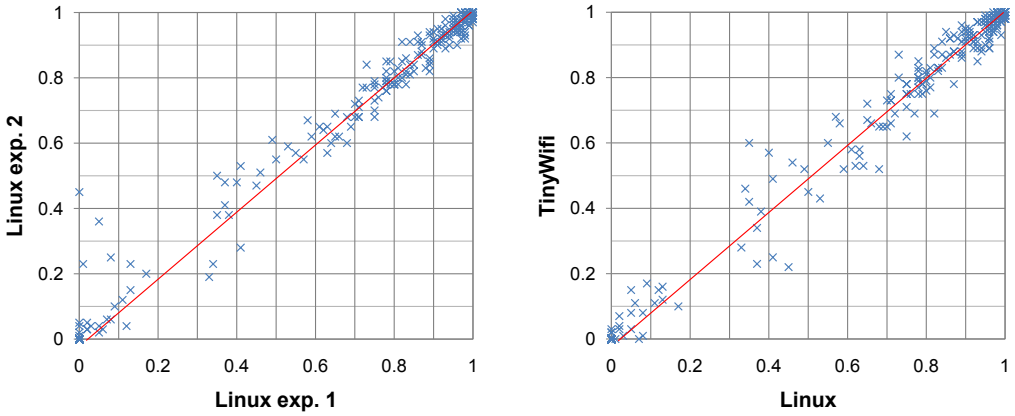
In this section we focus on evaluating the correctness of TinyWifi implementation by observing link and network layer behavior. Our evaluation aspects aim to demonstrate the correctness and versatility of TinyWifi rather than stress-testing the employed protocols or platforms.

Before evaluating complex protocols using TinyWifi, we stress-tested all its design features using test applications shipped with TinyOS. For example, applications like *Blink* and *BlinkToRadio* demonstrate the proper functioning of timers and radio communication, respectively. Similarly, *BlinkTask*, *Oscilloscope* and *Multihop-Oscilloscope* prove an accurate implementation of split-phase operation, sensors and the serial message interface.

We evaluate TinyWifi on UMIC-Mesh [ZGW+06] and Indriya [DCA09] testbeds. UMIC-Mesh is a Linux based meshnet deployed at RWTH Aachen University. It consists of 51 IEEE 802.11a/b/g based mesh-routers[2] located in various rooms at the department of computer science. Each node has a 500 MHz CPU and 256 MB of RAM. Indriya is a sensornet deployed of 127 nodes (cf. Section 5.4.1). Each node on Indriya has an MSP430 CPU with 10 KB of RAM and a low power CC2420 radio, which can run IEEE 802.15.4 protocols. The major characteristics[3] of these testbeds are shown in Table 6.1.3.

[2]Only 35 were available for our experiments

[3]We refer readers to the respective testbed websites for connectivity graphs and further information: http://www.umic-mesh.net/meshconf/#geographical and http://indriya.comp.nus.edu.sg/motelab/html/index.php



(a) PRRs for back-to-back experiments in Linux

(b) PRRs for TinyWifi vs Linux

Figure 6.5 Packet Reception Rates: PRR comparison between TinyWifi and Linux on IEEE 802.11. Each link PRR is estimated using a native Linux socket protocol as well as TinyWifi protocol. If both the protocols estimated that a given link is of the same quality, the point would lie on the 45 degree line. There are a total of 1226 points representing the PRR of each link in the network. Overall, TinyWifi and Linux native link estimation agree, hence most of the points are near the 45 degree line.

In the following we evaluate the correctness and applicability of TinyWifi both at the link and network layers and through the behavior of native TinyOS protocols in an IEEE 802.11 testbed.

6.1.3.1 Link Layer

We first show that the communication service of TinyWifi does not impact the behavior of the underlying link layer when compared to the native platform. To this end, we correlate the Packet Reception Rates (PRR) of nodes using both TinyWifi and the native platform, i.e., Linux. Ideally, in the scatter-plot representation of such a correlation, every single data point should lie on the 45 degree line. However, this is not even achieved in back-to-back experiments on the native Linux platform as shown in Figure 6.5(a). This is due to the unpredictable and highly dynamic nature of the wireless medium. Figure 6.5(b) depicts the correlation between PRRs of TinyWifi and Linux platforms. We can clearly observe the strong similarity between Figures 6.5(a) and 6.5(b). Hence, we conclude that the (user-space) implementation of TinyWifi does not adversely affect the link level behavior of the underlying radio technology.

6.1.3.2 Network Layer

To evaluate the correctness of TinyWifi on the network layer, we compare the behavior of a native TinyOS protocol in TinyWifi with the behavior of a Linux-native implementation of the same protocol. We show that the nesC implementation of

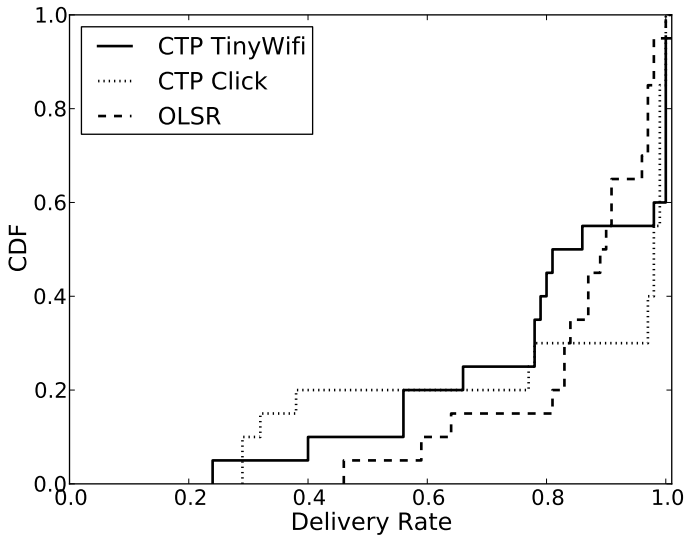


Figure 6.6 CDF of per node delivery rate: The performance of CTP under TinyWifi and Click is very comparable. The average delivery rate of CTP TinyWifi is 0.81, CTP Click achieves 0.82 and OLSR 0.85.

a protocol for TinyOS, when evaluated using TinyWifi, is commensurate with its native counterpart. To this end, we evaluate the behavior of the Collection Tree Protocol (CTP) in both its nesC and Click [KMC+00] implementations. Click is a highly recognized software architecture for building modular and configurable protocols.

Our comparative analysis uses the CTP protocol for several reasons: CTP has become a de-facto standard in collection routing in sensornets. It has also been implemented in various languages to support different OS and simulation platforms, such as Mantis OS, Contiki OS, Sun SPOTs, and Castalia Simulation. It has been thoroughly tested using six different MAC layers. The mechanisms used in CTP have also been incorporated in IETF RPL - the IPv6 protocol for low-power and lossy networks. Recognizing its highly efficient and reliable delivery in networks with lossy links, CTP has been extended for point-to-point communications in meshnets.

Delivery rate is a metric commonly used in evaluating sensornet protocols. It is equivalent to the average end-to-end reliability between sensor nodes and the receiver that receives the sensor data using multihop routing. Our key evaluation metric is the *delivery rate* for two reasons: (1) The current TinyWifi implementation is not optimized for throughput evaluations, and (2) the default operational parameters, such as buffer sizes, of the protocols and the platforms under consideration are different. To establish a fair comparison base for other performance benchmarks, such as throughput and jitter, we need to modify these parameters. However, this is beyond the scope of our contribution in this dissertation. In our experiment on UMIC-Mesh, we used one node as the destination in the network. All other nodes send a burst of 100 packets, one at a time to the single destination. The receiver node simply logs the received packets identified by a unique sequence number and a sender ID. To establish a baseline and to enable better understanding of the results, we also compare CTP with OLSR, a standard routing protocol for meshnets. Figure 6.6 shows the cumulative distribution of the delivery rates for both implementations of CTP and OLSR. Figure 6.7 displays the pairwise delivery rates for each node pair

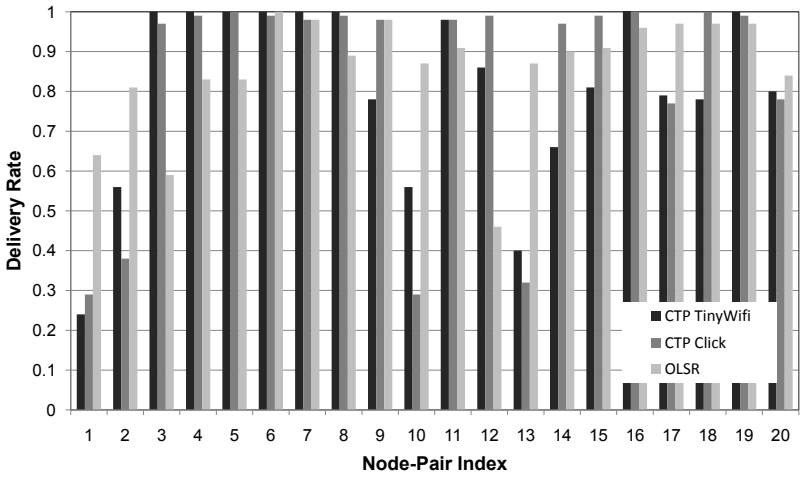


Figure 6.7 Delivery rates for each sender-receiver pair using OLSR and two implementations of CTP. For better understanding and visibility, the graph does not show the results for sender-receiver pairs with less than 0.15 delivery rate.

in detail. These results show that the performance of Linux native CTP (which required reimplementation) is similar to the TinyWifi version of CTP.

Overall, these results conclude that TinyWifi enables the direct and unaltered execution of nesC protocols in IEEE 802.11 based networks. The implementation overhead of TinyWifi modules does not influence protocol performance as shown in the case of CTP. The minor difference in the results (i.e., 1%) could be due to the varying link qualities across the experiments. This means that using TinyWifi, the implementation effort for the Click implementation of CTP [CJLBB11] (i.e., approx. 7000 lines of codes excluding Click libraries) could be saved.

6.1.4 Limitations

In its current implementation, TinyWifi serves as a general *enabling* platform for multiple link layers. Due to this focus and our effort to keep a small code-base, TinyWifi does not export specific link layer services of either the original or the target OS. Currently, this means that TinyOS protocols that rely on a specific link-layer service which is not provided by IEEE 802.11 link layers are not supported. As one example, the MultihopLQI collection protocol (released with TinyOS) heavily relies on *link quality indicator* (LQI) information for establishing routing tables. Hence, MultihopLQI will only be applicable in a meshnet if the corresponding link-layer exports LQI information to higher layer protocols. The support of specific link layer services means a tradeoff between the implementation complexity and the usefulness and applicability of this service in the target domain.

Similarly, TinyWifi, at the moment, only focusses on enabling TinyOS routing protocols to run in Linux. We see this as a first step towards a general platform for protocol and application experimentation and evaluation. However, some TinyOS characteristics, such as a one-packet outgoing buffer, remain because TinyOS protocols rely on and are designed for them. To make full use of a target platform's (additional) resources, we need a mechanism to allow protocols and applications to capitalize on these resources as well as they can.

It is widely believed that network layer protocols such as geographic routing, graph embedding [JS03], and AODV can work on a variety of wireless networks. TinyWifi enables the validation of such hypotheses. Our preliminary results suggest that some protocols (e.g., routing) do work on vastly different wireless networks - on radios that support a few tens of Kbps to those that support tens of Mbps data rates. However, we note that not all protocols can work in such widely different platforms. The network protocols running on IEEE 802.15.4 radios can assume constant bit rate while the network protocols running on IEEE 802.11b radios cannot make that assumption. Packet delivery takes a more predictable time on 802.15.4 radios compared to 802.11b radios due to more complex OS kernel, NIC driver, and generally higher level of programming interface. Some assumptions about link layer properties, although ideally avoided, are implicitly embedded in the network protocol design. TinyWifi helps us test the network protocols in vastly different radio platforms and discover such implicit assumptions behind network protocol design.

6.1.5 Related Work

EmStar [GRE+07] and VIPE [LKGW09] are two notable related efforts that enable network protocol execution across heterogenous computation and communication platforms.

EmStar is a software environment for deploying complex applications on heterogenous sensornet designs, incorporating a mixture of mote-class devices and Linux driven micro-servers. The idea is to leverage the additional resources of a distributed micro-server based network to improve robustness and system visibility of sensornet deployments. EmStar provides its own runtime environment, protocol execution on different classes of devices is not a goal of this approach. TinyWifi, on the other hand, aims at migrating the whole protocol to a completely different class of wireless networks.

VIPE evolves the implementation of a protocol from its design up to its deployment without re-implementation of single parts. It provides an encapsulation of minimal core functionality in small building blocks. These blocks may then be used by a protocol on different platforms (i.e., simulation, emulation, testbeds, and real-world deployments). TinyWifi is similar to VIPE in a sense that it provides a common service architecture across multiple platforms. However, VIPE assumes a common programming and runtime environment across these platforms. Unlike TinyWifi, it does not address the challenges associated with spanning a wider range of platforms and thus would require existing IEEE 802.15.4 based protocols to be re-implemented using VIPE's interface abstractions.

Building protocols for multiple networking classes is a common trade in today's systems. For example, DHCP operates on multiple link layers (e.g., Ethernet and WiFi) as well as wireless cards from different vendors. However, these link layers are highly standardized (with common interfaces and runtime environments), and span resource-rich platforms ranging from data centers to embedded systems. Hence, existing cross-platform protocols are restricted to very similar platforms. Besides saving re-implementation effort, the distinctive feature of TinyWifi is that it enables protocols to run across *vastly* different link layers (i.e., IEEE 802.11 and 802.15.4)

with even wider ranges of device capabilities: Going all the way from 8-bit microcontrollers, with a few KB of memory and low-power radios capable of data rates as low as a few tens of Kbps, to platforms with an order of magnitude higher processing and storage capabilities and equipped with radios that support data rates up to few tens of Mbps. TinyWifi elegantly addresses the associated challenges, such as a different programming and runtime environment, for bridging protocols between such a wider range of platforms.

Similarly, over the past few years, we have seen a common evolution of protocols and concepts (e.g., AODV [PBRD03] and ETX metric [DCABM05]) originally developed for MANETs and meshnets being modified and ported to sensornets. However, this evolution has generally followed the same trend and direction, i.e., from MANETs to sensornets. TinyWifi enables this new trend in protocol evolution: Protocols and concepts developed for sensornets can now be ported and evaluated in MANETs and meshnets.

6.2 Evaluating PAD in IEEE 802.11

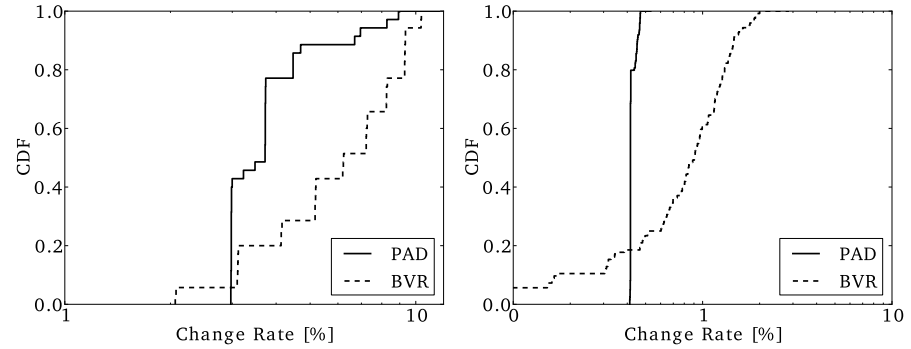
After presenting the design and evaluation of TinyWifi, we now focus on the evaluation of PAD in IEEE 802.11 based wireless networks, such as meshnets and MANETs. This also demonstrates the utility of TinyWifi as an efficient evaluation and runtime platform. We note that this evaluation only aims at establishing the basic understanding and providing initial insights into the generality and feasibility of our approaches in IEEE 802.11 based wireless networks. However, a detailed design space exploration for IEEE 802.11 is beyond the scope of this dissertation.

To this end, we first perform a similar comparative evaluation of PAD, as in Chapter 5, on UMIC meshnet. The goal here is to see if PAD carries its superior performance across different classes of wireless networks. Then we evaluate PAD from mobility perspective to see if it can be considered as a suitable candidate for routing in meshnets with mobile nodes. We use OMNeT++ simulator to create different mobility patterns for evaluating PAD's performance.

6.2.1 Testbed Evaluation

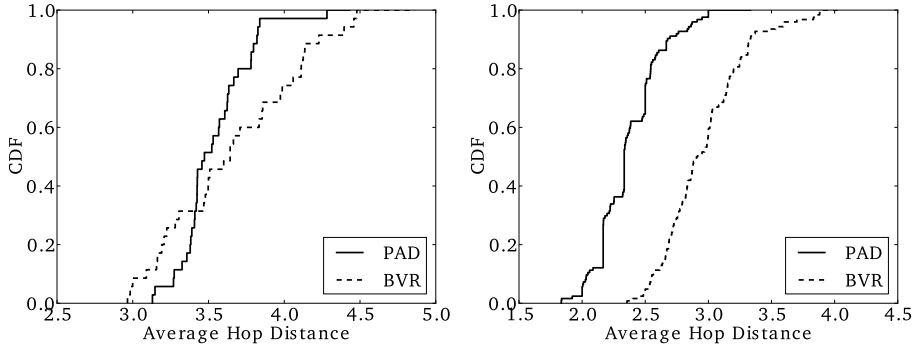
Our UMIC meshnet evaluation compares PAD with BVR [FRZ+05] and S4 [MWQ+10]. All these three protocols are native to sensornets (IEEE 802.15.4) but their mechanisms are believed to be equally relevant for meshnets and ad-hoc networks (IEEE 802.11) [AWK+11a]. However, until now, their applicability is only limited to sensornets due to (1) the significant re-implementation effort associated with porting these protocols to other classes of wireless networks, and (2) the lack of an integrated wireless development platform. TinyWifi enables evaluation of these protocols in IEEE 802.11 networks and thus provides a deeper insight into their behavior without re-implementation.

The key performance metrics of these protocols considered here include: (1) address stability, (2) average hop distance from landmarks, and (3) the number of transmissions required for a packet to reach its destination. Our evaluation for the first



(a) Coordinate change rate on 802.11 testbed

(b) Coordinate change rate on 802.15.4 testbed



(c) Average hop distance from landmarks on 802.11 testbed

(d) Average hop distance from landmarks on 802.15.4 testbed

Figure 6.8 Addressing results from IEEE 802.11 and 802.15.4 based testbeds. PAD maintains its superior performance in terms of address stability across multiple wireless network classes.

two metrics compares PAD with BVR only. This is because S4 shares the virtual coordinates establishment with BVR (cf. Section 5.4.3). The experimental setup is similar to the setup discussed in Section 5.4.1 except that we now additionally perform these experiments on UMIC-Mesh using TinyWifi.

6.2.1.1 Address Stability

Address updates are expensive in wireless networks where nodes have to determine their own addresses based on the underlying connectivity in the network. In such virtual coordinates based protocols, addresses are typically stored in a database. Frequent address changes thus result in a significant overhead due to frequent updates in the address database. Hence, address stability is one of the key performance measures of virtual coordinates based routing protocols. Figures 6.8(a) and 6.8(b) show the cumulative distribution of address change rate in IEEE 802.11 (UMIC) and IEEE 802.15.4 (Indriya) networks, respectively. The address change rate is defined as the share of routing update intervals in which the nodes update their addresses. These results clearly show that PAD performs better than BVR in IEEE 802.11 networks. However, the magnitude of improvement is smaller compared to what was observed in IEEE 802.15.4 networks. This is due to the different node degrees in the networks: PAD derives its addresses from multiple paths leading towards a

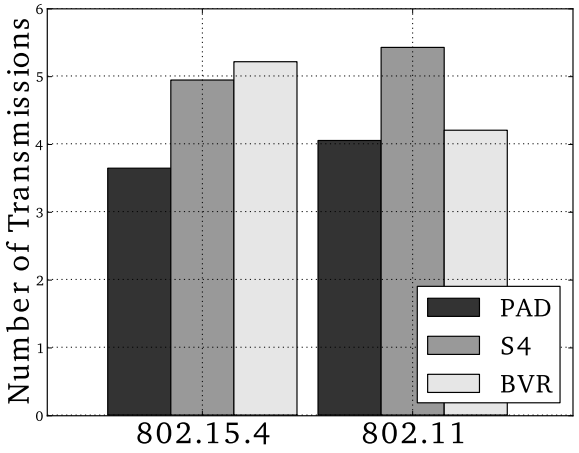


Figure 6.9 Routing results from IEEE 802.11 and 802.15.4 based testbeds. The results show a similar trend in both domains: S4's cluster based approach struggles in sparse network environments.

landmark and tolerates link quality changes along a specific path. Hence, in a sparse network, such as UMIC, there is only a limited number of unique paths that can be represented in a PAD address.

6.2.1.2 Hop Distance

The hop distance metric determines the number of hops between a node and all landmarks in the network. Figures 6.8(c) and 6.8(d) depict the CDF of hop distances averaged over all landmark trees. It can be seen that PAD achieves lower hop distances than BVR in both testbeds. This is because PAD always enables shortest paths to dominate its coordinate distributions (cf. Section 5.4.3.3). Whereas, BVR only selects good quality paths using PRR based link estimation. Hence, due to the higher node degree in Indriya, the probability of the shortest path being different than the stable path selected by BVR is much higher. This is because higher node degree results in increasing the number of paths over which a landmark can be reached. This phenomenon is reflected by the difference in the results of Figures 6.8(c) and 6.8(d).

6.2.1.3 Routing Cost

Finally, we evaluate the routing cost, i.e., the average number of transmissions required for a packet to reach its destination. Figure 6.9 shows that PAD outperforms both S4 and BVR in the IEEE 802.15.4 network. However, in the IEEE 802.11 network, PAD and BVR achieve similar results while still performing better than S4. These results show that S4's performance is dependent upon dense deployments and a stable network topology.

Overall, this evaluation demonstrates that PAD maintains its superior performance across multiple wireless network classes. Moreover, it also indicates that TinyWifi provides important hints about protocol performance on different link layers and in different network types. Hence, the feasibility of a protocol in different classes of

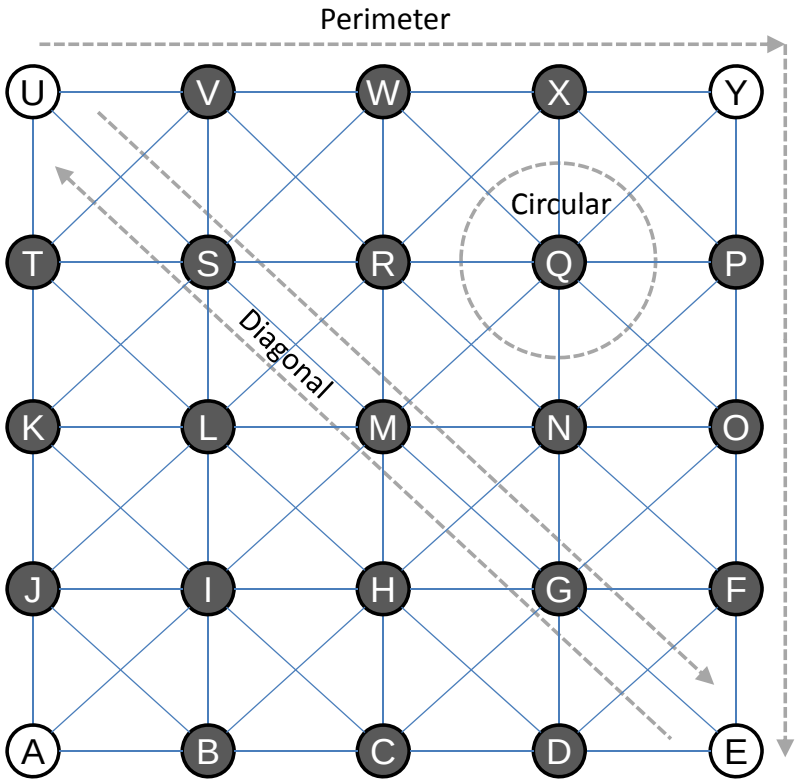


Figure 6.10 Simulation setup and mobility patterns.

wireless network cannot simply be assumed, it rather needs to be validated using platforms such as TinyWifi.

6.2.2 Evaluating PAD in Mobility

In this section we evaluate PAD in a meshnet with mobile nodes. A typical meshnet constitutes two types of nodes, *mesh-nodes* (or routers) and *mobile-clients*. Mesh nodes are static and form the basic infrastructure of a meshnet. These nodes provides services, such as Internet, and can communicate with each other possibly over multiple hops. Mobile-clients, on the other hand, are not permanent members of the network. These nodes can join the network to use a service and are free to move within the covered range of the meshnet.

One of the key challenges in meshnets is to support mobility while maintaining a high end-to-end delivery reliability. We believe that PAD can support mobility in meshnets because of its fuzzy addressing scheme that incorporates multiple paths leading towards a node in its address distribution. Therefore, we perform some basic experiments to asses the utility of PAD in such mobile environments.

6.2.2.1 Experimental Setup

Until now, our evaluation has been performed on real world testbeds to show the practical feasibility of the concepts presented in this dissertation. However, in this section, we use OMNeT++ simulator to evaluate PAD from mobility perspective.

This is because our primary focus here is to create network dynamics using different mobility patterns. A simulation environment, such as OMNeT++, allows us to create such mobility patterns which otherwise cannot easily be established or repeated in a static testbed environment.

We create a simple 5x5 grid link mesh-topology of static mesh nodes, as shown in Figure 6.10. Nodes A, E, U and Y are the designated landmarks. Using this topology as an underlying mesh-infrastructure, we add one mobile node to the network with the following mobility patterns (cf. Figure 6.10).

- **Circular:** The mobile nodes revolves around one mesh-node (i.e., node Q) in the network.
- **Perimeter:** The mobile node follows the perimeter of the network.
- **Diagonal:** The mobile node moves across the network forming a diagonal path.

Among the three mobility patterns, the *diagonal* is the most challenging since the movement of the node also impacts the virtual coordinates of the mesh-nodes. This is because the mobile node offers additional paths towards landmarks while moving through the network.

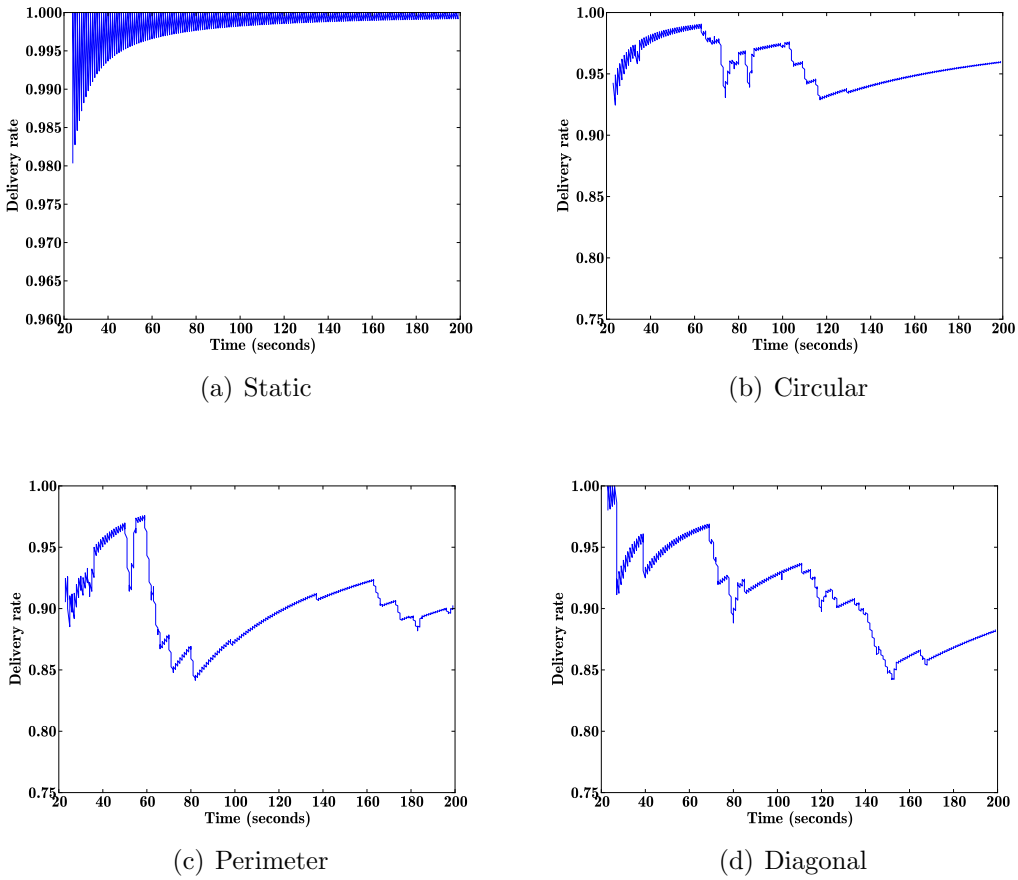
We did not employ any transport layer protocol, such as TCP, to ensure that the delivery rates of PAD are not influenced by the end-to-end retransmissions mechanisms of these protocols. Moreover, our experimental setup has the following key characteristic: (1) Node A acts as a sender in all the experiments. (2) Each node sends a beacon every 2 seconds. (3) The payload length is 800 bytes. And (4) each experiment lasts for 200 seconds, i.e., the time in which the mobile node completes one traversal of the path.

6.2.2.2 Results

Tables 6.2.2.1 summarizes the results for different mobility patterns. In order to establish a comparison base, we first performed an experiment to observe the delivery rate of PAD in a static meshnet scenario, i.e., when the client node is not moving. Figure 6.11(a) shows that PAD achieves a delivery reliability of ∼100% in the static network scenario. This result implies that, in the experiments for different mobility patterns, the incurred packet loss will be due to the network dynamics introduced

Property	Static	Around	Perimeter	Diagonal
Delivery rate	1	0.96	0.90	0.88
Average Hop-count	4	3.96	4.95	3.52
% packets in fallback-mode	0	1.8	3.75	1.5
Address change rate	3.48	8.58	12.50	12.42

Table 6.2 Summary of mobility results. PAD maintains a very high delivery rate for different mobility patterns. The address change rate represents the percentage of routing beacon intervals in which the node changes its address and requires an update in the address database.



(a) Static (b) Circular

(c) Perimeter (d) Diagonal

Figure 6.11 Evaluating PAD in meshnets with different mobility patterns. PADmaintains a very high delivery rate in all the cases.

by the mobile node. Moreover, the address change rate in the static scenario means that a node will have to change its address in only 3.48% of routing beacon intervals (cf. Section 5.4.3.1 for the definition of address change rate).

Figure 6.11 shows that PAD maintains a very high delivery rate in all the cases. We can clearly see that the *diagonal* mobility pattern is very challenging for virtual coordinate based addressing schemes. This is because the mobile client can simultaneously impact the coordinates of all the nodes in the network, and thereby the routing topology. Whereas in the *circular* and *perimeter* mobility patterns, the mobile client does not necessarily influence the coordinates of the nodes. For example, a neighboring node will only change its coordinates if the mobile client offers a shorter path towards a landmark than its current parent. This rarely happens for circular and perimeter mobility patterns.

Moreover, we can see a trend in the delivery rate for all the three mobility patterns: It is that there is a sudden drop in the delivery rate followed by a sustained improvement. For example, in the case of circular and perimeter mobility patterns, this drop occurs at time 120s and 80s, respectively. However, for diagonal mobility, this trends repeats more often. This sudden drop in the delivery rate occurs due to sudden address changes in the network triggered by the movement of mobile node. Similarly, the sustained improvement in delivery rate afterwards points to the quick recovery of PAD from such address dynamics in the network.

Overall, these results demonstrate the principle feasibility of PAD in meshnets and MANETs.

6.3 Summary

We presented TinyWifi, an evaluation platform for portable network experiments on different layer technologies. Using TinyWifi, developers can evaluate a single implementation of prototypes across multiple wireless network classes such as sensornets and meshnets. We demonstrated the utility of TinyWifi by evaluating four well known routing protocols in two testbeds that make use of different radio technologies. One key observation is that the communication service of TinyWifi does not impact the behavior of the underlying link layer. At the network layer, we observed that the nesC implementation of a protocol performed at par with native Linux implementation in Click. Moreover, our evaluation shows that TinyWifi allows us to better understand and reason about the performance characteristics of protocols in different networking environments.

Next to presenting TinyWifi, we explored the feasibility of PAD in IEEE 802.11 based networks. We observed that PAD carries its superior performance across different classes of wireless networks. We also evaluated PAD using different mobility patterns. The evaluation demonstrated the utility of PAD for networks with mobile nodes, such as meshnets and MANETs. The fuzzy addressing scheme in PAD requires very few address updates for different mobility patterns in the network.

Overall, this chapter concludes that network layer protocols, which do not rely on a specific link layer technology can work on a variety of wireless network classes. However, the true utility of these protocols in a different class of wireless network cannot simply be assumed, it needs to be evaluated using integrated platforms such as TinyWifi. As an example, we observed that a change in even a single network characteristic, such as node density, can significantly impact the performance of a protocol.

7

Discussion and Conclusions

In this dissertation we propose different approaches of link estimation, routing, and addressing to enhance multihop communications in wireless networks. The basic idea is to utilize long-range intermediate links for packet forwarding and thereby reduce the number of transmissions in the network. The inclusion of these links, however, requires relevant support at three different levels: First, we develop link estimation metrics to estimate the quality of these links at shorter time scales. Based on these metrics, we design a link estimator that identifies intermediate links with bursty characteristics. Second, we design appropriate routing extensions that facilitate the integration of the proposed link estimator into the routing infrastructure. Finally, we present a robust virtual coordinate based addressing scheme that exploits such links in point-to-point routing without compromising the stability of a node's coordinates.

All the three approaches reside at the network layer of the protocol stack and are not limited to one specific class of wireless networks: For example, our evaluation on both IEEE 802.15.4 and IEEE 802.11 based networks shows that the proposed mechanisms are independent of the underlying link layer technology. One key achievement is that our prototype implementations outperform the state-of-the-art in link estimation, routing, and addressing in sensornets. Besides thoroughly evaluating the proposed approaches on widely used testbeds, the data analysis presented in this dissertation provides even a greater depth of detail about the extent and applicability of the previously ignored class of wireless links.

The rest of this chapter is structured as follows. Section 7.1 summarizes the major contributions of this dissertation by revisiting the key concepts of link estimation, routing, and addressing. In Section 7.2, we discuss the lessons learnt during different phases of this work and shed light on the limitations. Finally, we highlight the future directions in Section 7.3.

7.1 Summary

In this section we summarize our main contributions by focusing on the key concepts discussed in the preceding chapters.

7.1.1 Link Estimation

Existing approaches of link estimation typically use PRR based link metrics. These metrics are calculated over a very long time period and thus only reveal long term characteristics of a link. However, we have seen that intermediate links are stable in the short term and can offer significantly better routing progress. Efficient utilization of such links for packet forwarding can therefore reduce the number of transmissions required by a packet to reach its destination.

Our observations from widely used testbeds, such as MoteLab and Mirage, show that more than 60% of the intermediate links are bursty. These links alternatively shift between short-term reliable and unreliable periods of transmissions. Therefore, we need specialized metrics to cope with the dynamics revealed by these links and to accurately identify reliable periods of transmissions on these links.

As a first step, we propose two metrics for estimating intermediate links, MAC_3 and EFT. MAC_3 estimates link burstiness at runtime. The goal of this metric is to differentiate links with correlated packet reception events from the links with independent transmission characteristics. Approaches such as the β-factor can measure link burstiness, however, as we observed in Chapter 3, β is not feasible for runtime link estimation. MAC_3 is a very simple metric and achieves high accuracy during runtime even when applied over a small transmission history of a link. We also compare MAC_3 with PRR. Our empirical evaluation reveals that MAC_3 can replace PRR as a link estimation metric since the value assigned by MAC_3 to a link is at least as good as its PRR value.

EFT complements MAC_3 by providing information about the length of successful transmission bursts over a link. The goal is to identify bursty links with smaller transmission bursts. This is important because links with smaller transmission bursts trigger rapid changes in the routing topology and thus can be detrimental for the routing performance.

Finally, we develop a link estimator, BLE, based on these metrics. BLE is a full fledged link estimator which performs all the typical link estimation operations such as link addition, deletion, and reinforcement. The evaluation results demonstrate that BLE accurately identifies bursty links in the network and render them available for transmission by discovering successful transmission periods over these links. Moreover, BLE works in parallel with existing long-term link estimators to identify long-range intermediate links which are typically ignored by today's link estimators and routing protocols.

Our major findings with regard to BLE's performance are summarized in Table 7.1.1.

Approaches Compared	Settling Time	Runtime Accuracy	Link Discovery
β-factor	unpredictable	not accurate	unknown
PRR	small	accurate	variable (depends on τ)
BLE	small ($<$ 100 packets)	accurate	after 3 packets

Table 7.1 Link estimation summary: The metric used by BLE has small settling time and achieves high accuracy when compared with its base value. It discovers available links as quickly as the next three packets.

7.1.2 Routing

Routing protocols employ link estimators to identify high quality links in the network for routing purposes. A traditional routing approach is to build routing trees where each node selects parent(s) among its neighbors that minimize(s) the number of transmissions towards the tree root. However, in doing so, today's routing protocols converge routing to only a few paths. This approach results in inefficient utilization of the network resources because a multitude of other, potentially valuable paths based on intermediate links remain unutilized.

We propose BRE to utilize such communication paths in the network. The main goal of BRE is to incorporate intermediate links into the routing process. For this purpose, it enables seamless integration of short term link estimators, such as BLE and STLE, with today's routing protocols.

BRE introduces two routing modes, *bursty* and *traditional*. In bursty mode, packets are forwarded over an intermediate link if it is currently reliable for transmission and offers higher routing progress. BRE rapidly falls back to traditional mode once an intermediate link is no longer available. Hence, it exploits communication opportunities that are typically ignored by routing protocols. However, it maintains a stable routing topology because it operates greedily, i.e., it does not propagate these local changes in link selection in the whole network. We believe that BRE strikes an efficient tradeoff between the stability of routing and its adaptability to the underlying link conditions.

The concept of BRE is a general one and is not tied to any specific protocol. It can be integrated with any routing protocol that uses PRR based link estimation metric. Our evaluation clearly demonstrates the utility of BRE for reducing the number of transmissions in the network. Our results from MoteLab show that it minimizes routing cost by up to 40%.

Table 7.1.2 summarizes BRE evaluation results.

Approaches Compared	Improvement (in %) using BRE		
	Transmissions	Throughput	Reliability
CTP	19%	7%	0%
StrawMan	63%	-6%	–

Table 7.2 Routing results summary. BRE reduces transmission count when compared with CTP and StrawMan. The use of BRE does not impact the end-to-end reliability of transmission. StrawMan improves routing throughput at a very cost - 63% increase in transmission count.

Approaches Compared	Improvement with PAD			
	Address Stability	Address Monotony	Transmission Count	Hop Count
BVR	7x	12x	35%	20%
S4	7x	–	26%	20%

Table 7.3 Routing results summary. BRE reduces transmission count when compared with CTP and StrawMan. The use of BRE does not impact the end-to-end reliability of transmission. StrawMan improves routing throughput at a very high cost - 63% increase in transmission count.

7.1.3 Addressing

BVR is a widespread approach of virtual coordinates based point-to-point routing in sensornets. It assigns virtual coordinates to nodes based on their hop distances to a small set of beacons. In BVR, the next hop towards a certain landmark, i.e., the address vector for that landmark, is greedily selected based on the long-term link quality. However, in a dynamic network, a node's address will frequently change due to recurrently changing distances to landmarks. In such a dynamic scenario, assigning static addresses to nodes often results in an inconsistent routing topology. Thereby, introducing significant overhead due to regular updates in the address database.

To overcome this limitation in virtual coordinate based routing protocols, we introduce PAD, a virtual coordinate system based on the statistical distribution of a node's distance from a set of landmarks. The basic idea is that a node learns from its past addresses and calculates a probability distribution of its address vectors. It then advertises this distribution to other nodes in the network instead of a static current address. All other nodes predict the current location of a node in its address distribution. For example, the simplest prediction would be to calculate the *mean* for each address vector component (i.e., the *mean* distance to each landmark). PAD, as it is based on the statistical distribution of the node's location in the network, eliminates the need to use expensive link estimation.

PAD facilitates the inclusion of intermediate links into the routing process by incorporating multiple paths leading towards a landmark in its distribution. A prototype implementation of PAD and a thorough evaluation with regard to address stability and routing performance demonstrate its superior performance over widespread routing protocols such as BVR and S4.

Table 7.1.3 summarizes the comparative evaluation of PAD, BVR, and S4.

7.1.4 Portable Protocol Evaluation

To evaluate the generality of the approaches presented in this dissertation, we developed TinyWifi, a TinyOS platform enabling convenient and robust support for Linux driven network devices. Due to the inherent similarities between sensornets and meshnets, communication protocols for TinyOS can now easily be evaluated in IEEE 802.11 based networks. TinyWifi integrates seamlessly into the existing TinyOS source and provides all the necessary hardware independent functionality.

TinyOS applications can therefore be compiled and executed on Linux driven network devices without modifications. This approach saves protocol re-implementation effort and allows researchers to evaluate their protocols across multiple classes of wireless networks.

During the development of TinyWifi, we faced several architectural and design challenges, such as the split-phase operations, that demanded sophisticated solutions in order to support such a programming paradigm in Linux. Similarly, we derive timing functionalities from only a single Linux timer and construct a fully functional radio communication interface.

Finally, using TinyWifi, we show the feasibility of PAD in IEEE 802.11 based wireless networks. PAD carries its superior performance over BVR and S4 even in a vastly different wireless networking environment.

7.2 Lessons Learnt

In this section we summarize our key findings and pinpoint the limitations of the approaches presented in this dissertation.

7.2.1 Intermediate Links are Bursty

Our analysis using data from Mirage and MoteLab shows that the majority of wireless links in these testbeds are bursty. However, the extent of burstiness is strongly dependent upon the time scale of measurement. At shorter time scales, links show higher correlation between packet loss events and one can predict the fate of future transmissions with high probability. Correspondingly, we observed that intermediate links are more beneficial when traffic patterns are bursty and when transmissions rates are higher. At slower transmission rates packet forwarding over such links does not offer any significant advantage over traditional routing approaches.

7.2.2 Bursty Links are Useful for Routing

BRE is the first approach that exercises unicast transmissions over an intermediate link which is completely ignored by today's link estimators and routing protocols. It shows that intermediate links are useful for reducing transmission count and increasing routing throughput.

However, packet overhearing based link estimation introduces additional overhead since a node has to receive and process packets which are not necessarily addressed to it. It also means that our approach is not directly applicable with duty-cycled MAC protocols (i.e., low power listening) where packet overhearing is prohibitive.

7.2.3 Nodes can be Addressed Probabilistically

We observed that introducing error tolerance and fuzziness in the form of a probabilistic address is a more suitable approach to achieve stable addressing in dynamic

network conditions. Using this approach, PAD offers an order of magnitude higher address stability and monotony then contemporary addressing techniques.

The major drawback, however, is the larger node addresses that increases the control packet overhead. Nonetheless, this additional control overhead is well compensated by avoiding expensive link estimation and packet overhearing mechanisms employed by today's routing protocols.

7.2.4 Network Protocols are Directly Portable

Finally, we observed that network layer protocols in sensornets can be directly ported to a different class of wireless network. We showed that PAD maintains it superior performance over BVR and S4 in both IEEE 802.15.4 and 802.11 based wireless networks.

However, this porting from sensornets to meshnets is not always possible, especially (1) when the protocol (e.g. MultihopLQI) relies on a specific link layer service (e.g. LQI) which is not available in IEEE 802.11, or (2) when the protocol assume a constant bit rate, or (3) when the protocol (e.g. time synchronization) relies on more predictable link level behavior such as packet delivery time.

7.3 Future Work

In the following we briefly highlight the possible future directions for this work.

7.3.1 Integrating BLE with Routing Protocols

Integrating BLE with PRR based routing protocols in different wireless domains — IEEE 802.11 and IEEE 802.15.4 — and a detailed evaluation of its performance benefits is ongoing work. Similarly, we are interested in evaluating the generality of our parameters, such as history size and error thresholds, and using more rigorous approaches for selecting these parameters. Moreover, our interest lies in understanding how BLE scales with different table sizes, node densities, topologies and traffic patterns.

7.3.2 Exploring BRE alternatives

After evaluating the effectiveness of BRE for routing over intermediate quality wireless links, we identify the following aspects as future work: (1) Classifying overhearing nodes based on their success history to avoid repeated selection of a node that did not offer significant improvement over the traditional path, (2) limiting link selection to the ones that offer at least one hop reduction to avoid even the rare occurrence of bad results, (3) Integrating BRE with low-power listening techniques, and (4) Implementing and comparing new algorithms for BRE, such as including path-ETX of the parent node in the packet headers to eliminate BRE's dependence on routing table size.

7.3.3 Routing Algorithms for PAD

We are still in the early phases of investigating suitable routing algorithms and distance functions, such as Gaussian distance, that can operate on PAD's addresses even more efficiently. Similarly, in mobile networks, a routing algorithm can exploit coordinate variances (i.e., variance in hop counts from a certain landmark) to predict a node's movement and improve delivery reliability.

Another important aspect is to analyze coordinate correlations to see if address changes at one node trigger similar changes in neighboring nodes. This would help us to improve our address prediction mechanism in that a node can expect a similar shift in neighboring node's coordinates by observing the shift in its own coordinate distribution. We are also interested in analyzing the patterns in PAD's addresses and define them in terms of a probability density function. We need to understand how PAD reacts to tuning different parameters like the number of landmarks in the network, the beacon interval length, and the routing table size. It would be of interest to see if PAD maintains its superior performance over BVR and other virtual coordinate based addressing mechanism after varying these parameters.

7.3.4 Evolving TinyWifi

Although our protocol evaluation demonstrates the correctness of TinyWifi implementation, we still need to stress-test different design components such as timers and split-phase operations. Besides using multi-hop routing protocols, we plan to expand our work to evaluating and supporting dissemination and network time synchronization protocols. Finding a well-balanced set of features that supports a multitude of protocols and applications as well as better providing the resources of the target platform to protocols are further steps in the development of TinyWifi.

Moreover, our evaluation on IEEE 802.11 based testbeds only compares PAD with sensornet protocols, i.e., BVR and S4. A thorough comparative evaluation with native meshnet protocols, such as OLSR and AODV, is important to establish a deep understanding and a broader relevance of PAD in wireless networking.

Bibliography

[ABB+04] Daniel Aguayo, John Bicket, Sanjit Biswas, Glenn Judd, and Robert Morris. Link-level measurements from an 802.11b mesh network. *SIGCOMM Comput. Commun. Rev.*, 34:121–132, August 2004.

[AGKL10] Muhammad Hamad Alizai, Lei Ga, Torsten Kempf, and Olaf Landsiedel. Tools and modeling approaches for simulating hardware and systems. In *Modeling and Tools for Network Simulation*. Springer - Lecture Notes in Computer Science, 2010.

[AKL+10] Muhammad Hamad Alizai, Bernhard Kirchen, Jó Ágila Bitsch Link, Hanno Wirtz, and Klaus Wehrle. Poster abstract: Tinyos meets wireless mesh networks. In *Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems*, SenSys '10, pages 429–430, New York, NY, USA, 2010. ACM.

[AKLW10] Muhammad Hamad Alizai, Georg Kunz, Olaf Landsiedel, and Klaus Wehrle. Promoting power to a first class metric in network simulations. In *Proc. of the Workshop on Energy Aware Systems, in conjunction with GI/ITG ARCS 2010*, 2010.

[ALBL+09] Muhammad Hamad Alizai, Olaf Landsiedel, Jó Agila Bitsch Link, Stefan Götz, and Klaus Wehrle. Routing over bursty wireless links. Technical report, Technical University Hamburg, 9 2009.

[ALL+09] Muhammad Hamad Alizai, Olaf Landsiedel, Jó Ágila Bitsch Link, Stefan Götz, and Klaus Wehrle. Bursty traffic over bursty links. In *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems*, SenSys '09, pages 71–84, New York, NY, USA, 2009. ACM.

[ALW07] Muhammad Hamad Alizai, Olaf Landsiedel, and Klaus Wehrle. Accurate timing in sensor network simulations. Technical report, RWTH Aachen, Aachen, Germany, 7 2007.

[ALW09] Muhammad Hamad Alizai, Olaf Landsiedel, and Klaus Wehrle. Modelling execution time and energy consumption in sensor node simulation. *PIK Journal, Special Issue on Energy Aware Systems*, 32(2), 2009.

[ALW12] Muhammad Hamad Alizai, Olaf Landsiedel, and Klaus Wehrle. Exploiting the burstiness of intermediate quality wireless links. *International Journal of Distributed Sensor Networks (IJDSN)*, 826702, 2012.

[ALWB08] Muhammad Hamad Alizai, Olaf Landsiedel, Klaus Wehrle, and Alexander Becher. Challenges in short-term wireless link quality estimation. In *Proceedings of the 7th GI/ITG Fachgespraech Wireless Sensor Networks (FGSN'08)*, Berlin, Germany, 2008.

[AVL+10] Muhammad Hamad Alizai, Tobias Vaegs, Olaf Landsiedel, Raimondas Sasnauskas, and Klaus Wehrle. Poster abstract: Statistical vector based point-to-point routing in wireless networks. In *IPSN'10*, 2010.

[AVL+11] Muhammad Hamad Alizai, Tobias Vaegs, Olaf Landsiedel, Stefan Goetz, Jo Agila Bitsch Link, and Klaus Wehrle. Probabilistic addressing: Stable addresses in unstable wireless networks. In *IPSN 2011: Proceedings of the 10th ACM/IEEE International Conference on Information Processing in Sensor Networks*, 2011.

[AWK+11a] Muhammad Hamad Alizai, Hanno Wirtz, Benhard Kirchen, Tobias Vaegs, Omprakash Gnawali, and Klaus Wehrle. Tinywifi: Making network protocol evaluation portable across multiple phy-link layers. In *Proceedings of the Sixth ACM International Workshop on Wireless Network Testbeds, Experimental evaluation and Characterization (WiNTECH), Las Vegas, NV, USA*, 9 2011.

[AWK+11b] Muhammad Hamad Alizai, Hanno Wirtz, Georg Kunz, Benjamin Grap, and Klaus Wehrle. Efficient online estimation of bursty wireless links. In *16th IEEE Symposium on Computers and Communications (ISCC), Kerkyra, Greece*, IEEE, 6 2011.

[Bec07] Alexander Becher. Design and implementation of an adaptive point-to-point routing protocol for wireless sensor networks. Diploma Thesis, RWTH Aachen University, 2007.

[BLKW08] Alexander Becher, Olaf Landsiedel, Georg Kunz, and Klaus Wehrle. Towards short-term link quality estimation. In *Proceedings of the international workshop on Hot topics in embedded networked sensing systems*, HotEmnets '08, 2008.

[BM05a] Sanjit Biswas and Robert Morris. Exor: opportunistic multi-hop routing for wireless networks. *SIGCOMM Comput. Commun. Rev.*, 35:133–144, August 2005.

[BM05b] Sanjit Biswas and Robert Morris. Exor: opportunistic multi-hop routing for wireless networks. In *Proceedings of the 2005 conference on Applications, technologies, architectures, and protocols for computer communications*, SIGCOMM '05, pages 133–144, New York, NY, USA, 2005. ACM.

[BMSU99] Prosenjit Bose, Pat Morin, Ivan Stojmenović, and Jorge Urrutia. Routing with guaranteed delivery in ad hoc wireless networks. In *Proceedings of the 3rd international workshop on Discrete algorithms and methods for mobile computing and communications*, DIALM '99, pages 48–55, New York, NY, USA, 1999. ACM.

[BY09] Jonathan Billington and Cong Yuan. On modelling and analysing the dynamic manet on-demand (dymo) routing protocol. *T. Petri Nets and Other Models of Concurrency*, 3:98–126, 2009.

[CA06] Qing Cao and Tarek Abdelzaher. Scalable logical coordinates framework for routing in wireless sensor networks. *ACM Trans. Sen. Netw.*, 2:557–593, November 2006.

[CAAKA10] Z. Che-Aron, W. F. M. Al-Khateeb, and F. Anwar. The enhanced fault-tolerant aodv routing protocol for wireless sensor network. In *Proceedings of the 2010 Second International Conference on Computer Research and Development*, ICCRD '10, pages 105–109, Washington, DC, USA, 2010. IEEE Computer Society.

[CBA+05] B. N. Chun, P. Buonadonna, A. Auyoung, Chaki Ng, D. C. Parkes, J. Shneidman, A. C. Snoeren, and A. Vahdat. Mirage: a microeconomic resource allocation system for sensornet testbeds. In *Proceedings of the 2nd IEEE Workshop on Embedded Networked Sensors (EmNets)*, May 2005.

[CBPG11] Rémi Chou, Yvo Boers, Martin Podt, and Matthieu Geist. Performance evaluation for particle filters. In *14th International Conference on Information Fusion (FUSION 2011)*, Chicago, USA, 2011. IEEE. to appear.

[Chi02] Chipcon. *CC1000 Datasheet: Single Chip Very Low Power RF Transceiver*, April 2002.

[CJ03] T. Clausen and P. Jacquet. Optimized Link State Routing Protocol (OLSR). RFC 3626, IETF, October 2003.

[CJLBB11] Jung Il Choi Choi, Mayank Jain, Jung Woo Lee, and Juan Batiz-Benet, 2011. CLICK CTP, Stanford Information Networking Group: http://sing.stanford.edu/gnawali/ctp/.

[CM02] Benjie Chen and Robert Morris. L+: Scalable landmark routing and address lookup for multi-hop wireless networks. *Massachusetts Inst. Technol. (MIT) Tech. Rep.*, 2002.

[CWK+05a] Alberto Cerpa, Jennifer L. Wong, Louane Kuang, Miodrag Potkonjak, and Deborah Estrin. Statistical model of lossy links in wireless sensor networks. In *Proceedings of the 4th international symposium on Information processing in sensor networks*, IPSN '05, Piscataway, NJ, USA, 2005. IEEE Press.

[CWK+05b] Alberto Cerpa, Jennifer L. Wong, Louane Kuang, Miodrag Potkonjak, and Deborah Estrin. Statistical model of lossy links in wireless sensor networks. In *IPSN*, 2005.

[CWPE05] Alberto Cerpa, Jennifer L. Wong, Miodrag Potkonjak, and Deborah Estrin. Temporal properties of low power wireless links: Modeling

and implications on multi-hop routing. In *Proc. of the 6th ACM Interational Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, May 2005.

[DA10] Rajesh Deshmukh and Asha Ambhaikar. Performance evaluation of aodv and dsr with reference to network size. *International Journal of Computer Applications*, 11(8):27–32, December 2010. Published By Foundation of Computer Science.

[DAG03] Murat Demirbas, Anish Arora, and Mohamed G. Gouda. A pursuer-evader game for sensor networks. In *Proceedings of the 6th international conference on Self-stabilizing systems*, SSS'03, pages 1–16, Berlin, Heidelberg, 2003. Springer-Verlag.

[Dai] J. Daintith. *Oxford Dictionary of Chemistry*. Oxford University Press.

[Dat09] SING Datasets, 2009. Stanford Information Networking Group.

[DCA09] Manjunath Doddavenkatappa, Mun Choon Chan, and Ananda A.L. An experience of building indriya. In *National University of Singapore*, 2009.

[DCABM05] Douglas S. J. De Couto, Daniel Aguayo, John Bicket, and Robert Morris. A high-throughput path metric for multi-hop wireless routing. *Wirel. Netw.*, 11:419–434, July 2005.

[Dev08] TinyOS Developers. Tinyos 2.1 adding threads and memory protection to tinyos. In *Proceedings of the 6th ACM conference on Embedded network sensor systems*, SenSys '08, pages 413–414, New York, NY, USA, 2008. ACM.

[DPZ04] Richard Draves, Jitendra Padhye, and Brian Zill. Comparison of routing metrics for static multi-hop wireless networks. In *SIGCOMM '04: Proceedings of the 2004 conference on Applications, technologies, architectures, and protocols for computer communications*, volume 34, pages 133–144, New York, NY, USA, October 2004. ACM.

[DVB01] Michel Denuit and Sebastien Van Bellegem. On the stop-loss and total variation distances between random sums. *Statistics & Probability Letters*, 53(2):153–165, June 2001.

[EFK07] Jakob Eriksson, Michalis Faloutsos, and Srikanth V. Krishnamurthy. Dart: dynamic address routing for scalable ad hoc and mesh networks. *IEEE/ACM Trans. Netw.*, 15:119–132, February 2007.

[ERS06] Cheng Tien Ee, Sylvia Ratnasamy, and Scott Shenker. Practical data-centric storage. In *Proceedings of the 3rd conference on Networked Systems Design & Implementation - Volume 3*, NSDI'06, pages 24–24, Berkeley, CA, USA, 2006. USENIX Association.

[EV07] Sinem Coleri Ergen and Pravin Varaiya. Energy efficient routing with delay guarantee for sensor networks. *Wirel. Netw.*, 13(5), 2007.

[FGJL07] Rodrigo Fonseca, Omprakash Gnawali, Kyle Jamieson, and Philip Levis. Four-bit wireless link estimation. In *Sixth Workshop on Hot Topics in Networks (HotNets)*, November 2007.

[FRZ+05] Rodrigo Fonseca, Sylvia Ratnasamy, Jerry Zhao, Cheng Tien Ee, David Culler, Scott Shenker, and Ion Stoica. Beacon vector routing: scalable point-to-point routing in wireless sensornets. In *Proceedings of the 2nd conference on Symposium on Networked Systems Design & Implementation - Volume 2*, NSDI'05, pages 329–342, Berkeley, CA, USA, 2005. USENIX Association.

[GE00] Govindan and D. Estrin. Directed diffusion: a scalable and robust communication paradigm for sensor networks. In *Proceedings of the sixth annual international conference on Mobile computing and networking*, 2000.

[GEH03] Deepak Ganesan, Deborah Estrin, and John Heidemann. Dimensions: why do we need a new data handling architecture for sensor networks? *SIGCOMM Comput. Commun. Rev.*, 33:143–148, January 2003.

[GFJ+09] Omprakash Gnawali, Rodrigo Fonseca, Kyle Jamieson, David Moss, and Philip Levis. Collection tree protocol. In *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems*, SenSys '09, pages 1–14, New York, NY, USA, 2009. ACM.

[GGSE01] Deepak Ganesan, Ramesh Govindan, Scott Shenker, and Deborah Estrin. Highly-resilient, energy-efficient multipath routing in wireless sensor networks. *SIGMOBILE Mob. Comput. Commun. Rev.*, 5(4), 2001.

[GLC05] David Gay, Phil Levis, and David Culler. Software design patterns for tinyos. In *Proceedings of the 2005 ACM SIGPLAN/SIGBED conference on Languages, compilers, and tools for embedded systems*, LCTES '05, pages 40–49, New York, NY, USA, 2005. ACM.

[GLvB+03a] David Gay, Philip Levis, Robert von Behren, Matt Welsh, Eric Brewer, and David Culler. The nesc language: A holistic approach to networked embedded systems. In *Proceedings of the ACM SIGPLAN 2003 conference on Programming language design and implementation*, PLDI '03, pages 1–11, New York, NY, USA, 2003. ACM.

[GLvB+03b] David Gay, Philip Levis, Robert von Behren, Matt Welsh, Eric Brewer, and David Culler. The nesc language: A holistic approach to networked embedded systems. *SIGPLAN Not.*, 38:1–11, May 2003.

[Gra10] Benjamin Grap. Identifying reliable opportunistic neighbors in multihop wireless networks. Bachelor Thesis, RWTH Aachen University, 2010.

[GRE+07] Lewis Girod, Nithya Ramanathan, Jeremy Elson, Thanos Stathopoulos, Martin Lukac, and Deborah Estrin. Emstar: A software environment for developing and deploying heterogeneous sensor-actuator networks. *ACM Trans. Sen. Netw.*, 3, August 2007.

[GSAP06] C. Gomez, P. Salvatella, O. Alonso, and J. Paradells. Adapting aodv for ieee 802.15.4 mesh sensor networks: Theoretical discussion and performance evaluation in a real environment. In *Proceedings of the 2006 International Symposium on on World of Wireless, Mobile and Multimedia Networks*, WOWMOM '06, pages 159–170, Washington, DC, USA, 2006. IEEE Computer Society.

[Har00] John Harris. *An Introduction to Fuzzy Logic Applications*. Kluwer Academic Publishers, Norwell, MA, USA, 2000.

[Hed88] C. L. Hedrick. Routing information protocol, 1988.

[HKS+04] Tian He, Sudha Krishnamurthy, John A. Stankovic, Tarek Abdelzaher, Liqian Luo, Radu Stoleru, Ting Yan, Lin Gu, Jonathan Hui, and Bruce Krogh. Energy-efficient surveillance system using wireless sensor networks. In *Proceedings of the 2nd international conference on Mobile systems, applications, and services*, MobiSys '04, pages 270–283, New York, NY, USA, 2004. ACM.

[HKWW06] Vlado Handziski, Andreas Köpke, Andreas Willig, and Adam Wolisz. Twist: a scalable and reconfigurable testbed for wireless indoor experiments with sensor networks. In *REALMAN*, 2006.

[HP09] Min Li Huang and Sin-Chong Park. A wlan and zigbee coexistence mechanism for wearable health monitoring system. In *Proceedings of the 9th international conference on Communications and information technologies*, ISCIT'09, pages 555–559, Piscataway, NJ, USA, 2009. IEEE Press.

[HPH+05] Vlado Handziski, Joseph Polastre, Jan-Hinrich Hauer, Cory Sharp, and Adam Woliszand David Culler. Flexible hardware abstraction for wireless sensor networks. In *Proceedings of the Second European Workshop on Wireless Sensor Networks (EWSN '05)*, 2005.

[HSNW10] Islam Hegazy, Reihaneh Safavi-Naini, and Carey Williamson. Towards securing mintroute in wireless sensor networks. In *Proceedings of the 2010 IEEE International Symposium on A World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, WOWMOM '10, pages 1–6, Washington, DC, USA, 2010. IEEE Computer Society.

[HSW+00] Jason Hill, Robert Szewczyk, Alec Woo, Seth Hollar, David Culler, and Kristofer Pister. System architecture directions for networked sensors. *SIGPLAN Not.*, 35:93–104, November 2000.

[IN99] Tomasz Imieliński and Julio C. Navas. Gps-based geographic addressing, routing, and resource discovery. *Commun. ACM*, 42:86–92, April 1999.

[Ins07] Texas Instruments. *CC2420 Datasheet: Single-Chip 2.4 GHz IEEE 802.15.4 Compliant and ZigBee(TM) Ready RF Transceiver*, March 2007.

[JM96] David B. Johnson and David A. Maltz. Dynamic source routing in ad hoc wireless networks. In *Mobile Computing*, pages 153–181. Kluwer Academic Publishers, 1996.

[JS03] Newso James and Dawn Song. Gem: Graph embedding for routing and data-centric storage in sensor networks without geographic information. In *Proceedings of the 1st international conference on Embedded networked sensor systems*, SenSys '03, pages 76–88, New York, NY, USA, 2003. ACM.

[JSX02] C. Jiao, L. Schwiebert, and B. Xu. On modeling the packet error statistics in bursty channels. In *Proceedings of the 27th Annual IEEE Conference on Local Computer Networks*, LCN '02, pages 0534–, Washington, DC, USA, 2002. IEEE Computer Society.

[KCPnC09] Ankur Kamthe, Miguel Á. Carreira-Perpiñán, and Alberto E. Cerpa. M&m: multi-level markov model for wireless link simulations. In *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems*, SenSys '09, pages 57–70, New York, NY, USA, 2009. ACM.

[KHC09] Tae Hyun Kim, Jae Yeol Ha, and Sunghyun Choi. Improving spectral and temporal efficiency of collocated ieee 802.15.4 lr-wpans. *IEEE Transactions on Mobile Computing*, 8:1596–1609, 2009.

[Kir10] Bernhard Kirchen. Tinywifi: Enabling linux platform support in tinyos. Bachelor Thesis, RWTH Aachen University, 2010.

[KK00] Brad Karp and H. T. Kung. Gpsr: greedy perimeter stateless routing for wireless networks. In *Proceedings of the 6th annual international conference on Mobile computing and networking*, MobiCom '00, pages 243–254, New York, NY, USA, 2000. ACM.

[Kle08] Alexander Klein. Performance comparison and evaluation of AODV, OLSR and SBR in mobile ad-hoc networks. In *International Symposium on Wireless Pervasive Computing (ISWPC)*, page 5, Santorini, Greece, May 2008.

[KMC+00] Eddie Kohler, Robert Morris, Benjie Chen, John Jannotti, and M. Frans Kaashoek. The click modular router. *ACM Trans. Comput. Syst.*, 18:263–297, August 2000.

[KS06] Kyu-Han Kim and Kang G. Shin. On accurate measurement of link quality in multi-hop wireless mesh networks. In *Proceedings of the 12th annual international conference on Mobile computing and networking*, MobiCom '06, pages 38–49, New York, NY, USA, 2006. ACM.

[KWZZ03] Fabian Kuhn, Roger Wattenhofer, Yan Zhang, and Aaron Zollinger. Geometric ad-hoc routing: of theory and practice. In *Proceedings of the twenty-second annual symposium on Principles of distributed computing*, PODC '03, pages 63–72, New York, NY, USA, 2003. ACM.

[Lan06] K.G. Langendoen. Apples, oranges, and testbeds. In *3rd IEEE Conf. on Mobile Ad-hoc and Sensor Systems (MASS 2006)*, pages 367–396, Vancouver, Canada, oct 2006.

[LAW08] Olaf Landsiedel, Hamad Alizai, and Klaus Wehrle. When timing matters: Enabling time accurate and scalable simulation of sensor network applications. In *Proceedings of the 7th international conference on Information processing in sensor networks*, IPSN '08, pages 344–355, Washington, DC, USA, 2008. IEEE Computer Society.

[LC11] Tao Liu and Alberto Eduardo Cerpa. Foresee (4c): Wireless link prediction using link features. In *Proceedings of the 10th international conference on Information processing in sensor networks*, IPSN '11, New York, NY, USA, 2011. ACM.

[LCL07] HyungJune Lee, Alberto Cerpa, and Philip Levis. Improving wireless simulation through noise modeling. In *Proceedings of the 6th international conference on Information processing in sensor networks*, IPSN '07, pages 21–30, New York, NY, USA, 2007. ACM.

[LG09] Philip Levis and David Gay. *TinyOS Programming*. Cambridge University Press, New York, NY, USA, 1st edition, 2009.

[LKC06] Seoung-Bumn Lee, Kyung J. Kwak, and A. T. Campbell. Solicitation-based forwarding for sensor networks. In *Sensor and Ad Hoc Communications and Networks, 2006. SECON '06. 2006 3rd Annual IEEE Communications Society on*, volume 1, pages 90–99, 2006.

[LKGH03] Xin Li, Young Jin Kim, Ramesh Govindan, and Wei Hong. Multidimensional range queries in sensor networks. In *Proceedings of the 1st international conference on Embedded networked sensor systems*, SenSys '03, pages 63–75, New York, NY, USA, 2003. ACM.

[LKGW09] Olaf Landsiedel, Georg Kunz, Stefan Götz, and Klaus Wehrle. A virtual platform for network experimentation. In *Proceedings of the 1st ACM workshop on Virtualized infrastructure systems and architectures*, VISA '09, pages 45–52, New York, NY, USA, 2009. ACM.

[LL08] Kaisen Lin and Philip Levis. Data discovery and dissemination with dip. In *Proceedings of the 7th international conference on Information processing in sensor networks*, IPSN '08, pages 433–444, Washington, DC, USA, 2008. IEEE Computer Society.

[LLM06] Ben Leong, Barbara Liskov, and Robert Morris. Geographic routing without planarization. In *Proceedings of the 3rd conference on Networked Systems Design & Implementation - Volume 3*, NSDI'06, pages 25–25, Berkeley, CA, USA, 2006. USENIX Association.

[LLWC03] Philip Levis, Nelson Lee, Matt Welsh, and David Culler. Tossim: accurate and scalable simulation of entire tinyos applications. In *Proceedings of the 1st international conference on Embedded networked sensor systems*, SenSys '03, pages 126–137, New York, NY, USA, 2003. ACM.

[LMG+04] Philip Levis, Sam Madden, David Gay, Joseph Polastre, Robert Szewczyk, Alec Woo, Eric Brewer, and David Culler. The emergence of networking abstractions and techniques in tinyos. In *NSDI'04: Proceedings of the 1st conference on Symposium on Networked Systems Design and Implementation*, 2004.

[LPCS04] Philip Levis, Neil Patel, David Culler, and Scott Shenker. Trickle: a self-regulating algorithm for code propagation and maintenance in wireless sensor networks. In *Proceedings of the 1st conference on Symposium on Networked Systems Design and Implementation - Volume 1*, pages 2–2, Berkeley, CA, USA, 2004. USENIX Association.

[LW07] Konrad Lorincz and Matt Welsh. Motetrack: a robust, decentralized approach to rf-based location tracking. *Personal Ubiquitous Comput.*, 11:489–503, August 2007.

[MALW10] Waqaas Munawar, Muhammad Hamad Alizai, Olaf Landsiedel, and Klaus Wehrle. Dynamic tinyos: Modular and transparent incremental code-updates for sensor networks. In *ICC'10: Proc. of the IEEE International Conference on Communications*, 2010.

[MOWW04] Thomas Moscibroda, Regina O'Dell, Mirjam Wattenhofer, and Roger Wattenhofer. Virtual coordinates for ad hoc and sensor networks. In *Proceedings of the 2004 joint workshop on Foundations of mobile computing*, DIALM-POMC '04, pages 8–16, New York, NY, USA, 2004. ACM.

[MRBT08] Andreas Meier, Tobias Rein, Jan Beutel, and Lothar Thiele. Coping with unreliable channels: Efficient link estimation for low-power wireless sensor networks. In *Proc. 5th Intl Conf. Networked Sensing Systems (INSS 2008)*, pages 19–26, Kanazawa, Japan, June 2008. IEEE.

[MSKG10] Scott Moeller, Avinash Sridharan, Bhaskar Krishnamachari, and Omprakash Gnawali. Routing without routes: the backpressure collection protocol. In *Proceedings of the 9th ACM/IEEE International Conference on Information Processing in Sensor Networks*, IPSN '10, pages 279–290, New York, NY, USA, 2010. ACM.

[MWQ+10] Yun Mao, Feng Wang, Lili Qiu, Simon Lam, and Jonathan Smith. S4: small state and small stretch compact routing protocol for large static wireless networks. *IEEE/ACM Trans. Netw.*, 18:761–774, June 2010.

[Nee] Michael J. Neely. Intelligent packet dropping for optimal energy-delay tradeoffs in wireless downlinks. In *Modeling and Optimization in Mobile, Ad-Hoc and Wireless Networks*, pages 179–188.

[Nee08] Michael J. Neely. Order optimal delay for opportunistic scheduling in multi-user wireless uplinks and downlinks. *IEEE/ACM Transactions on Networking*, 16:1188–1199, 2008.

[NGSA04] Suman Nath, Phillip B. Gibbons, Srinivasan Seshan, and Zachary R. Anderson. Synopsis diffusion for robust aggregation in sensor networks. In *SenSys '04: Proceedings of the 2nd international conference on Embedded networked sensor systems*, pages 250–262, 2004.

[Nic07] Dragoş Niculescu. Interference map for 802.11 networks. In *Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*, IMC '07, pages 339–350, New York, NY, USA, 2007. ACM.

[NW05] Hung T. Nguyen and Elbert A. Walker. *A First Course in Fuzzy Logic, Third Edition*. Chapman & Hall/CRC, 2005.

[OBM+07] Jorge Ortiz, Chris R. Baker, Daekyeong Moon, Rodrigo Fonseca, and Ion Stoica. Beacon location service: a location service for point-to-point routing in wireless sensor networks. In *Proceedings of the 6th international conference on Information processing in sensor networks*, IPSN '07, pages 166–175, New York, NY, USA, 2007. ACM.

[PB94] Charles E. Perkins and Pravin Bhagwat. Highly dynamic destination-sequenced distance-vector routing (dsdv) for mobile computers. In *Proceedings of the conference on Communications architectures, protocols and applications*, SIGCOMM '94, pages 234–244, New York, NY, USA, 1994. ACM.

[PBRD03] C. Perkins, E. Belding-Royer, and S. Das. Ad hoc on-demand distance vector (aodv) routing, 2003.

[PH06] Daniele Puccinelli and Martin Haenggi. Multipath fading in wireless sensor networks: measurements and interpretation. In *Proceedings of the 2006 international conference on Wireless communications and mobile computing*, IWCMC '06, pages 1039–1044, New York, NY, USA, 2006. ACM.

[PH08a] D. Puccinelli and M. Haenggi. Arbutus: Network-Layer Load Balancing for Wireless Sensor Networks. In *IEEE Wireless Communications and Networking Conference (WCNC'08)*, Las Vegas, NV, March 2008. Available at `http://www.nd.edu/~mhaenggi/pubs/wcnc08.pdf`.

[PH08b] Daniele Puccinelli and Martin Haenggi. Duchy: Double cost field hybrid link estimation for low-power wireless sensor networks. In *Proceedings of Fifth Workshop on Embedded Networked Sensors (Hot EmNets'08)*. ACM, June 2008.

[pLN10] Chih ping Li and Michael J. Neely. Energy-optimal scheduling with dynamic channel acquisition in wireless downlinks. *IEEE Transactions on Mobile Computing*, 9:527–539, 2010.

[PRD99] Charles E. Perkins, Elizabeth M. Royer, and Samir R. Das. Ad hoc on-demand distance vector (aodv) routing. In *2nd IEEE Workshop on Mobile Computing Systems and Applications*, 1999.

[PSC05] Joseph Polastre, Robert Szewczyk, and David Culler. Telos: enabling ultra-low power wireless research. In *Proceedings of the 4th international symposium on Information processing in sensor networks*, IPSN '05, Piscataway, NJ, USA, 2005. IEEE Press.

[QZW+07] Lili Qiu, Yin Zhang, Feng Wang, Mi Kyung Han, and Ratul Mahajan. A general model of wireless interference. In *Proceedings of the 13th annual ACM international conference on Mobile computing and networking*, MobiCom '07, pages 171–182, New York, NY, USA, 2007. ACM.

[Rap01] Theodore Rappaport. *Wireless Communications: Principles and Practice*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 2nd edition, 2001.

[REWT11] Christian Renner, Sebastian Ernst, Christoph Weyer, and Volker Turau. Prediction accuracy of link-quality estimators. In *Proceedings of the 8th European conference on Wireless sensor networks*, EWSN'11, pages 1–16, Berlin, Heidelberg, 2011. Springer-Verlag.

[RGJ+06] Fonseca Rodrigo, Omprakash Gnawali, Kyle Jamieson, Sukun Kim, Philip Levis, and Alec Wo. The collection tree protocol. In *TinyOS Enhancement Proposal, TEP 123*, August 2006.

[RL09] Tal Rusak and Philip Levis. Burstiness and scaling in the structure of low-power wireless links. *SIGMOBILE Mob. Comput. Commun. Rev.*, 13(1), 2009.

[RRP+03] Ananth Rao, Sylvia Ratnasamy, Christos Papadimitriou, Scott Shenker, and Ion Stoica. Geographic routing without location information. In *Proceedings of the 9th annual international conference on Mobile computing and networking*, MobiCom '03, pages 96–108, New York, NY, USA, 2003. ACM.

[RSBA07a] Krishna Ramachandran, Irfan Sheriff, Elizabeth Belding, and Kevin Almeroth. Routing stability in static wireless mesh networks. In *Proceedings of the 8th international conference on Passive and active network measurement*, PAM'07, pages 73–83, Berlin, Heidelberg, 2007. Springer-Verlag.

[RSBA07b] Krishna Ramachandran, Irfan Sheriff, Elizabeth Belding, and Kevin Almeroth. Routing stability in static wireless mesh networks. In Steve Uhlig, Konstantina Papagiannaki, and Olivier Bonaventure, editors, *Passive and Active Network Measurement*, volume 4427 of *Lecture Notes in Computer Science*, chapter 8, pages 73–82. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007.

[SA08] Weilian Su and Mohamad Alzaghal. Channel propagation measurement and simulation of micaz mote. *W. Trans. on Comp.*, 7:259–264, April 2008.

[SDTL06a] Kannan Srinivasan, Prabal Dutta, Arsalan Tavakoli, and Philip Levis. Some implications of low power wireless to ip networking. In *Proceedings of the 5th Workshop on Hot Topics in Networks (HotOS V)*, 2006.

[SDTL06b] Kannan Srinivasan, Prabal Dutta, Arsalan Tavakoli, and Philip Levis. Understanding the causes of packet delivery success and failure in dense wireless sensor networks. In *Proceedings of the 4th international conference on Embedded networked sensor systems*, SenSys '06, pages 419–420, New York, NY, USA, 2006. ACM.

[SJC+10] Kannan Srinivasan, Mayand Jain, Jung Il Choi, Tahir Azim, Edward S Kim, Philip Levis, and Bhaskar Krishnamachari. The κ-factor: Inferring protocol performance using inter-link reception correlation. In *Proceedings of the 16th Annual International Conference on Mobile Computing and Networking (Mobicom 2010)*. ACM, 2010.

[SKAL08] Kannan Srinivasan, Maria A. Kazandjieva, Saatvik Agarwal, and Philip Levis. The β-factor: measuring wireless link burstiness. In *Proceedings of the 6th ACM conference on Embedded network sensor systems*, SenSys '08, pages 29–42, New York, NY, USA, 2008. ACM.

[SKB10] Jorge Sá Silva, Bhaskar Krishnamachari, and Fernando Boavida, editors. *F-LQE: A Fuzzy Link Quality Estimator for Wireless Sensor Networks*, volume 5970 of *Lecture Notes in Computer Science*. Springer, 2010.

[SPMC04] Robert Szewczyk, Joseph Polastre, Alan M. Mainwaring, and David E. Culler. Lessons from a sensor network expedition. In *EWSN*, 2004.

[Sri10] Kannan Srinivasan. *Towards a Wireless Lexicon*. PhD thesis, Stanford University, 2010.

[Sta] SING Stanford. Stanford wireless access network (swan). `http://sing.stanford.edu/swan/`.

[SWWJ08] Hemant Sengar, Haining Wang, Duminda Wijesekera, and Sushil Jajodia. Detecting voip floods using the hellinger distance. *IEEE Transactions on Parallel and Distributed Systems*, 19:794–805, 2008.

[Tan96] K. Tanaka. *An introduction to Fuzzy Logic for practical applications*. Springer-Verlag, New York, 1996.

[Tan02] Andrew Tanenbaum. *Computer Networks*. Prentice Hall Professional Technical Reference, 4th edition, 2002.

[TE92] L. Tassiulas and A. Ephremides. Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks. *Automatic Control, IEEE Transactions on*, 37(12):1936–1948, 1992.

[Tec07] Crossbow Technology. Mica2: Data sheet. `http://www.xbow.com/Products/productdetails.aspx?sid=174`, July 2007.

[Tsu88a] P. F. Tsuchiya. The landmark hierarchy: a new hierarchy for routing in very large networks. *SIGCOMM Comput. Commun. Rev.*, 18:35–42, August 1988.

[Tsu88b] P. F. Tsuchiya. The landmark hierarchy: a new hierarchy for routing in very large networks. In *Symposium proceedings on Communications architectures and protocols*, SIGCOMM '88, pages 35–42, New York, NY, USA, 1988. ACM.

[Vae10] Tobias Vaegs. Statistical vector-based routing protocol for wireless sensor networks. Diploma Thesis, RWTH Aachen University, 2010.

[VAW10] Tobias Vaegs, Muhammad Hamad Alizai, and Klaus Wehrle. Probabilistic addressing in wireless networks. In *IEEE Student Conference, Hamburg, Germany*, 2010.

[vdM77] E. van der Meulen. A survey of multi-way channels in information theory: 1961-1976. *Information Theory, IEEE Transactions on*, 23(1):1–37, 1977.

[VW10] Tobias Vaegs and Klaus Wehrle. Poster: A statistical vector based routing protocol for wireless networks. In Hans K. Kaiser and Raimund Kirner, editors, *Proceedings of the Junior Scientist Conference 2010, Vienna, Austria*, pages 279–280, Karlsplatz 13, 1040 Vienna, Austria, 4 2010. Vienna University of Technology.

[WASW05] Geoffrey Werner-Allen, Patrick Swieskowski, and Matt Welsh. Motelab: a wireless sensor network testbed. In *Proceedings of the 4th international symposium on Information processing in sensor networks*, IPSN '05, Piscataway, NJ, USA, 2005. IEEE Press.

[WCL+07] Megan Wachs, Jung Il Choi, Jung Woo Lee, Kannan Srinivasan, Zhe Chen, Mayank Jain, and Philip Levis. Visibility: a new metric for protocol design. In *SenSys '07: Proceedings of the 5th international conference on Embedded networked sensor systems*, pages 73–86. ACM, 2007.

[WTC03] Alec Woo, Terence Tong, and David Culler. Taming the underlying challenges of reliable multihop routing in sensor networks. In *SenSys '03: Proceedings of the 1st international conference on Embedded networked sensor systems*, pages 14–27. ACM, 2003.

[WTV+07] Georg Wittenburg, Kirsten Terfloth, Freddy López Villafuerte, Tomasz Naumowicz, Hartmut Ritter, and Jochen Schiller. Fence monitoring: experimental evaluation of a use case for wireless sensor networks. In *Proceedings of the 4th European conference on Wireless sensor networks*, EWSN'07, pages 163–178, Berlin, Heidelberg, 2007. Springer-Verlag.

[WWAL+05] Matt Welsh, Geoff Werner-Allen, Konrad Lorincz, Omar Marcillo, Jeff Johnson, Mario Ruiz, and Jonathan Lees. Sensor networks for high-resolution monitoring of volcanic activity. In *Proceedings of the twenti-*

eth ACM symposium on Operating systems principles, SOSP '05, pages 1–13, New York, NY, USA, 2005. ACM.

[YIM+08] Tao Yang, Makoto Ikeda, Giuseppe De Marco, Leonard Barolli, Arjan Durresi, and Fatos Xhafa. Routing efficiency of aodv and dsr protocols in ad-hoc sensor networks. In *Proceedings of the 2008 The 28th International Conference on Distributed Computing Systems Workshops*, pages 66–71, Washington, DC, USA, 2008. IEEE Computer Society.

[YLRT09] Jiaxi You, Dominik Lieckfeldt, Frank Reichenbach, and Dirk Timmermann. Context-aware geographic routing for sensor networks with routing holes. In *Proceedings of the 2009 IEEE conference on Wireless Communications & Networking Conference*, WCNC'09, pages 2589–2594, Piscataway, NJ, USA, 2009. IEEE Press.

[Zan97] J. Zander. Radio resource management in future wireless networks: requirements and limitations. *Communications Magazine, IEEE*, 35(8):30 –36, aug. 1997.

[ZAS09] Hongwei Zhang, Anish Arora, and Prasun Sinha. Link estimation and routing in sensor network backbones: Beacon-based or data-driven? *IEEE Transactions on Mobile Computing*, 8:653–667, May 2009.

[ZG03] Jerry Zhao and Ramesh Govindan. Understanding packet delivery performance in dense wireless sensor networks. In *Proceedings of the 1st international conference on Embedded networked sensor systems*, SenSys '03, pages 1–13, New York, NY, USA, 2003. ACM.

[ZGW+06] Alexander Zimmermann, Mesut Günes, Martin Wenig, Jan Ritzerfeld, and Ulrich Meis. Architecture of the hybrid mcg-mesh testbed. In *Proceedings of the 1st international workshop on Wireless network testbeds, experimental evaluation & characterization*, WiNTECH '06, pages 88–89, New York, NY, USA, 2006. ACM.

[ZZHZ10] Ting Zhu, Ziguo Zhong, Tian He, and Zhi-Li Zhang. Exploring link correlation for efficient flooding in wireless sensor networks. In *Proceedings of the 7th USENIX conference on Networked systems design and implementation*, NSDI'10, pages 4–4, Berkeley, CA, USA, 2010. USENIX Association.

List of Figures

List of Tables

Nomenclature

4BLE	Four Bit Link Estimator
4C	Foresee: Wireless Link Prediction
BRE	Bursty Routing Extensions
MAC_3	Moving Average Conditional packet delivery function
PAD	Probabilistic Addressing
AODV	Ad hoc On-demand Distance Vector
ATF	Accept-to-Forward
BLE	Bursty Link Estimator
BVR	Beacon Vector Routing
CTP	Collection Tree Protocol
DART	Dynamic Address RouTing
DNH	Designated-next-Hop
EFT	Expected Future Transmissions
ETX	Expected Transmission Count
EWMA	Exponentially Weighted Moving Average
F-LQE	Fuzzy Link Quality Estimator
FPDF	Future Packet Delivery Function
HoPS	Holistic Packet Statistics
LTLE	Long Term Link Estimation
LTLE	Long Term Link Estimation
PRR	Packet Reception Rate
RSSI	Received Signal Strength Indication
S4	Small State Small Stretch Routing

SOFA	SOlicitation based ForwArding
STF	Solicit-to-Forward
STLE	Short Term Link Estimation
STLE	Short Term Link Estimation
WMEWMA	Window Mean Exponentially Weighted Moving Average

Index