A Probabilistic Framework for Multi-agent Distributed Interpretation and Optimization of Communication

Yang Xiang

Department of Computer Science
University of Regina
Regina, Saskatchewan, Canada S4S 0A2
E-mail: yxiang@cs.uregina.ca

September 9, 1994

Abbreviated title: Probabilistic Framework for Multi-agent Systems

Abstract

Multiply sectioned Bayesian networks for single-agent systems are extended into a framework for multi-agent distributed interpretation systems. Each agent is represented as a Bayesian subnet. Unlike in single-agent systems where evidence is entered one subnet at a time, multiple agents may acquire evidence asynchronously in parallel. New communication operations are thus proposed to maintain global consistency.

Inter-agent 'distance' prevents constant maintenance of global consistency. We show that, if new operations are followed, between two successive communications, answers to queries from an agent are consistent with all local evidence, and are consistent with all global evidence gathered up to the first communication.

During communication, each agent is not available to process evidence for a period of time (called *off-line time*). Two criteria for minimization of off-line time, which may commonly be used, are considered. We derive, under each criterion, the optimal schedules when communication is initiated from an arbitrarily selected agent.

1 Introduction

Probabilistic reasoning in Bayesian networks (BNs), as commonly applied, assumes a single-agent paradigm. That is, a single processor accesses a single global network representation, updates the joint probability distribution over the network variables as evidence becomes available, and answers queries. Concurrency, as applied to BNs, primarily aims at performance and decentralization of control [10, 4], but not at modeling inference among multiple agents with multiple perspectives. The resultant individual concurrent element is thus 'fine-grained', e.g., a node in a BN [10] or a clique in the junction tree representation of a BN [4].

The single-agent paradigm is inadequate when uncertain reasoning is performed by elements of a system between which there is some 'distance', which may be spatial, temporal, or semantic (elements are specialized differently) [1]. Such systems pose special issues that must be addressed. A multi-agent view is thus required where each agent is an autonomous intelligent subsystem. Each agent holds its own partial domain knowledge, accesses some external information source, and consumes some computational resource. Each agent communicates with other agents to achieve the system's goal cooperatively.

Distributed artificial intelligence (DAI) addresses the problems of designing and analyzing such 'large-grained' coordinating multi-agent systems [2, 3]. Main stream approaches in DAI, e.g., blackboard systems, contract nets, and open systems are essentially logic-based. To our best knowledge, little has been reported to explore probabilistic approach in DAI.

This paper reports our pilot study of applying probabilistic approach to distributed multi-agent reasoning. Our representation is based on multiply sectioned Bayesian networks (MSBNs)[12], which were developed for single-agent-oriented, modular knowledge

representation and more efficient inference[11]. We show that the modularity of MSBNs allows a natural extension into a multi-agent reasoning framework. We propose communication operations that are used to maintain inter-agent consistency. We derive communication schedules that optimize the time efficiency for communication.

We address the use of MSBNs in distributed interpretation, a subclass of problems in DAI. As defined originally by Lesser and Erman [7], an *interpretation* system accepts evidence from some environment and produces higher level descriptions of objects and events in the environment. A *distributed* interpretation system is needed when sensors for collecting evidence are distributed, and communication of all evidence to a centralized site is undesirable. Examples of such systems include sensor networks, medical diagnosis by multiple specialists, trouble-shotting of complex artifacts and distributed image interpretation.

Section 2 briefly introduce BNs and single-agent MSBNs. Section 3 presents the semantic extension of single-agent MSBNs to multi-agent MSBNs. Each cooperative agent is represented as a Bayesian subnet that consumes its own computational resource, gathers its own evidence, and can answer queries. Section 4 discusses consistency-related issues raised due to the extension, which are addressed in the subsequent two sections.

Unlike in single-agent systems where evidence is entered one subnet at a time, multiple agents may acquire evidence asynchronously in parallel. Section 5 addresses this issue and adds new belief propagation operations to the set of single-agent MSBN operations for inter-agent communication.

Section 6 proves that inter-agent consistency is guaranteed when the proposed operations are performed. Due to the inter-agent 'distance' and the associated communication cost, inter-agent consistency can not be maintained constantly. We prove that, when the proposed operations are performed, between two successive communications, the answers to queries from an agent are consistent with all local evidence gathered so far, and are consistent with all global evidence gathered up to the first communication of the two.

During communication, each agent is not available to process new evidence for a period of time (called *off-line time*). Such non-availability imposes restriction on time-critical applications. Therefore, the length of off-line time should be minimized. Section 7 defines two minimization criteria for off-line time which may commonly be used. One is based on the total length of off-line time for the entire multi-agent system. The other is based on the average length of off-line time across all agents in the system. To facilitate the study of the optimal communication schedules, we abstract the activities during communication into a graphical model, and identify the factors of communication that can be manipulated in optimizing schedules.

Section 8 derives the communication schedules that yield the minimal total length of off-line time, when communication is initiated from an arbitrarily selected agent.

Section 9 establishes the duality of the two major components of communication, relative to the off-line time study. Making use of duality, Section 10 derives the communication schedules that yield the minimal average off-line time, when communication is

initiated from an arbitrarily selected agent.

Section 11 discusses some general issues related to this work. Our presentation assumes a general terminology of graph theory.

2 Multiply Sectioned Bayesian Networks

2.1 Bayesian networks

A BN [10, 8, 6, 4] is a triplet (N, E, P). N is a set of nodes. Each node is labeled with a variable associated with a space. We shall use 'node' and 'variable' interchangeably. E is a set of arcs such that $D = (N, E)$ is a directed acyclic graph (DAG). The arcs signify the existence of direct causal influences between the linked variables. For each node $A_i \in N$, the strengths of the causal influences from its parent nodes π_i are quantified by a conditional probability distribution $p(A_i|\pi_i)$ of A_i conditioned on the values of A_i's parents. The basic dependency assumption embedded in BNs is that a variable is independent of its non-descendants given its parents. P is a joint probability distribution (JPD). For a BN with α nodes, P can be specified by the following product due to the assumption: $P = p(A_1 \ldots A_\alpha) = \prod_{i=1}^{\alpha} p(A_i|\pi_i)$.

2.2 Single Agent Oriented MSBNs

This subsection briefly introduces single-agent oriented MSBNs. Without confusion, we will refer to MSBNs only. For a formal presentation of MSBNs, see [12].

A MSBN consists of a set of interrelated Bayesian subnets. Each subnet represents one subdomain in a large domain. Each subnet shares a non-empty set of variables with at least one other subnet. The interfacing set between each pair of subnets must satisfy a *d-sepset* condition such that, when the pair is isolated from the MSBN, the interfacing set renders the two subnets conditionally independent.

The overall structure with which the subnets of a MSBN are organized is subject to a constraint called *soundness of sectioning*. Without the soundness of sectioning, a MSBN is subject to loss of information when later transformed into its secondary representation: a linked junction forest. A sufficient condition for sound sectioning is to construct a MSBN with a *hypertree* structure. The hypernodes in the hypertree are subnets of the MSBN. The hyperlinks are interfacing sets of nodes between subnets. Conceptually, the hypertree structured MSBN is built by adding one subnet (hypernode) at a time to existing ones. Only one interfacing set (hyperlink) to an existing subnet is explicitly stored. A hypertree structured MSBN guarantees that each subnet renders the rest of the MSBN conditionally independent.

Example 1 Figure 1 depicts a hypertree structured MSBN. Each box represents a subnet. Boundaries between boxes represent interfacing sets. The superscripts of subDAGs indicate a possible order of construction.

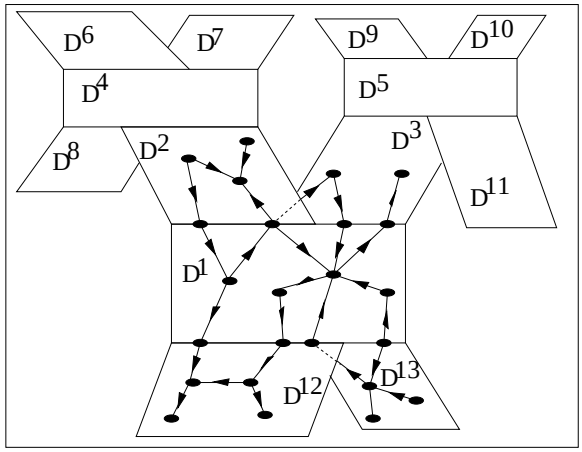

Figure 1: A MSBN with a hypertree structure.

Once a hypertree structured MSBN is constructed, it is then converted into a *linked junction forest* (LJF) of the identical hypertree structure. Each hypernode in the new hypertree is a *junction tree* (JT) (clique tree, join tree) converted from the corresponding subnet in the hypertree structured MSBN. The conversion of a subnet to a JT is subject to a *separability* constraint to guarantee non-distorted belief propagation. Each hyperlink of the new hypertree is a set of *linkages* converted from the corresponding interfacing set in the hypertree structured MSBN.

Parallel to the transformation of the graphical structure, there is a corresponding transformation of the probability tables in the MSBN to *belief tables* in the LJF. The conversion is subject to a *supportiveness* constraint such that belief can be propagated without blocking. The overall conversion guarantees that a *joint system belief* of the LJF, equivalent to the joint probability distribution of the MSBN, can be assembled from belief tables distributed in the LJF.

To answer queries by efficient local computation in a LJF, the LJF must be made consistent. A LJF is *locally consistent* if all JTs are internally consistent, i.e., when marginalized onto the same set of variables, different belief tables in a JT yield the identical marginal distribution. A LJF is *boundary consistent* if each pair of linked JTs are consistent with respect to their interfacing set. A LJF is *globally consistent* iff it is both locally consistent and boundary consistent.

A set of operations are developed to achieve consistency in a LJF during evidential reasoning: **BeliefInitialization** establishes initial global consistency. **DistributeEvidence** causes an outward belief propagation within a JT, and brings the JT internally consistent after evidence on variables in a single clique has been entered. **CollectEvidence** causes an inward belief propagation within a JT. **UnifyBelief** brings a JT internally consistent after evidence on variables in multiple cliques has been entered. **EnterEvidence** updates

belief in a JT in light of new evidence, and brings the JT internally consistent again by calling either **DistributeEvidence** or **UnifyBelief**. **UpdateBelief** updates the belief of a JT T relative to an adjacent JT, and brings T internally consistent. **DistributeBelief** initiated at a JT T causes an outward belief propagation in the LJF radiating from T. **ShiftAttention** allows the user to enter multiple pieces of evidence into a JT of current attention, and, when the user shifts attention to a target JT, maintains consistency along the hyperpath in the hypertree structured forest from the current JT to the target.

3 Representing Multiple Agents in MSBNs

As described in Section 2, a MSBN represents a large domain by representing a set of subdomains. From the viewpoint of reasoning agents, a MSBN represents multiple perspectives of a single agent. For example, PAINULIM [11] consists of three subnets which represents a neurologist's three different perspectives of the neuromuscular diagnostic domain: clinical, EMG and nerve conduction perspectives.

In a multi-agent system, each agent can be considered as holding its own perspective of the domain. This partial perspective may be over a subdomain, over a period of time, or over a spatial area. The modular representation of MSBN thus allows a natural extension to multi-agent systems, with a modification of the semantics: Instead of representing *one* agent's *multiple* perspectives of a domain, a multi-agent MSBN represents *multiple* agents in a domain each of which holds *one* distinct perspective of the domain. Each subnet corresponds to one such perspective.

We extend the example by Lauritzen and Spiegelhalter [6] to illustrate this:

Example 2 Dyspnoea (δ) may be due to tuberculosis (τ), lung cancer (ι) or bronchitis (β). A recent visit to Asia (ν) increases the chances of τ, while smoking (ζ) is a known risk factor for both τ and ι. After an initial diagnosis based on the above information, to further discriminate between τ and ι, a clinician may request lab tests from a radiology lab and a biology lab. Radiology lab has two relevant tests for τ and ι: X-ray (χ) and laminagraphy (α). Biology lab has two relevant tests as well: sputum test (ρ) and biopsy (o). Lab reports describe their impression upon τ and ι based on the results of the test(s).

The above fictitious example involves three reasoners: a clinical doctor, a radiologist, and a biologist. Diagnosis of a patient with dyspnoea is their common goal. But each reasoner has its own perspective of the same patient. The distance between them is semantic, and may be spatial as well.

A multi-agent MSBN can be constructed as a decision aid in such a domain. The MSBN consists of three agents: clinical, radiological, and biological subnets. The three agents may process evidence in parallel. Though it makes sense that the radiologist and biologist may not be involved until the clinician's initial diagnosis has been reached, and the patient has to visit them in sequence, considering the time delay required to develop

lab results, the radiological and biological subnets may well receive evidence and be posed with queries at the same time.

The resultant multi-agent MSBN is illustrated in Figure 2. The transformed LJF is shown in Figure 3. Note that, for this example, only a single linkage is created between a pair of JTs. In general, there may be multiple linkages between a pair of JTs (see the example in Xiang et al.[12] and the PAINULIM system in Xiang et al.[11]). Note also that, if the distance between the three agents are spatial, belief propagation must be performed over a communication channel, e.g., a computer network.

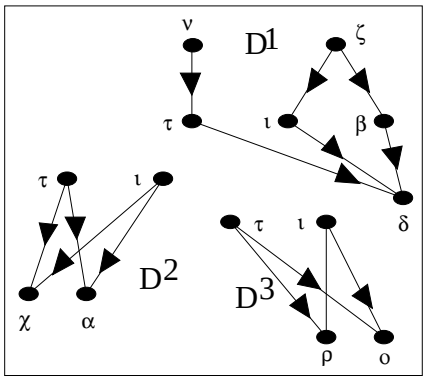

Figure 2: A multi-agent MSBN representing three medical specialists diagnosing a patient with dyspnoea. D^1: clinical subnet, D^2: radiological subnet, D^3: biological subnet.

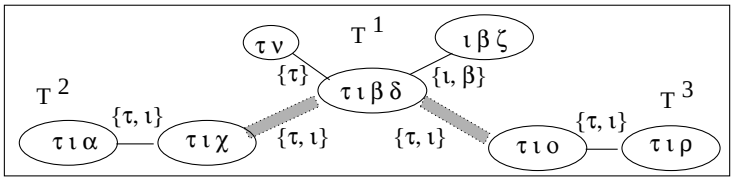

Figure 3: The LJF for the multi-agent MSBN in Figure 2. Sepsets between cliques are shown in solid lines. Linkages between JTs are shown in dotted bands.

4 Consistency Issues in Multi-agent MSBNs

The natural extension of MSBNs to multi-agent systems implies that all the technical constraints applicable to the construction of single-agent MSBNs must be followed in the construction of multi-agent MSBNs. In addition, evidential reasoning in multi-agent systems raises new issues regarding consistency, which must be addressed. To appreciate the issues, we first review how consistency is maintained in single-agent MSBNs:

Initial global consistency is obtained by **BeliefInitialization** (Section 2). The user focuses attention on one subdomain at a time. Therefore, evidence is entered *one subnet at*

a time. As the user shifts attention to each subnet and enters evidence, **ShiftAttention** (Section 2) maintains (local) consistency along the hyperpath in the hypertree structured forest. It is proven [12] that such local consistency is actually at the global level, i.e., answers to queries at the current JT is consistent with the evidence acquired in the entire LJF.

4.1 How to regain global consistency?

Note that **ShiftAttention** being able to maintain local consistency at the global level depends directly on the fact that evidence is always entered at the current subnet and nowhere else. In a multi-agent system, multiple agents may acquire evidence asynchronously in parallel. **ShiftAttention** can no longer be used to maintain global consistency. It must be replaced by new operations which we shall refer to as *communication*. We propose communication operations in Section 5 and prove their properties in Section 6.

4.2 What is the consistency level between communications?

Note also that **ShiftAttention** directly maintains only local consistency. But the local consistency happens to be at the global level in a single-agent MSBN! However, in a multi-agent MSBN, local consistency at a JT means one thing, and the global consistency means another. We can no longer shot two birds with one stone.

Even with new communication operations, due to the inter-agent 'distance' and the associated communication cost, global consistency can not be maintained constantly. A question that must be answered is, between two successive communications which regain global consistency, what is the consistency level of each agent after acquiring additional evidence? We prove a theorem to answer this question in Section 6.

5 Added Operations for Regaining Global Consistency

As discussed in Section 4, parallel evidence entering at multiple agents renders invalid the single-agent MSBN operation used to maintain global consistency. In order to regain global consistency in a multi-agent LJF, we extend the inward-outward belief propagation method in a single junction tree [4, 5] to a LJF. Jensen et al's method propagates belief through a single information path (a unique path exists between any two cliques in a JT). Belief propagation in a LJF must be performed over multiple linkages. Fortunately, the latter problem has been solved in single-agent MSBNs with the operation **UpdateBelief** (Section 2).

Following the above idea, we add two new operations **CollectNewBelief**[1] and **CommunicateBelief**[2] to regain global consistency in a multi-agent LJF. **CollectNewBelief** causes an inward belief propagation in the LJF. **CommunicateBelief** calls **CollectNewBelief** and **DistributeBelief** to propagate evidence obtained from multiple agents (JTs) inward first and then outward to the entire LJF.

Operation 3 (CollectNewBelief) *Let T be a JT in a LJF. Let* **caller** *be either the LJF or a neighbour JT. When* **CollectNewBelief** *is called in T, the following are performed:*

1. *T calls* **CollectNewBelief** *in all neighbours except* **caller** *if* **caller** *is a neighbour.*
2. *After each neighbour being called has finished* **CollectNewBelief**, *T updates its belief with respect to the neighbour by* **UpdateBelief**.

CollectNewBelief *is associated with JTs.*

Operation 4 (CommunicateBelief) *When* **CommunicateBelief** *is initiated at a LJF F, the following are performed:*

1. *A JT T in F is arbitrarily selected.*
2. **CollectNewBelief** *is called in T.*
3. *When T has finished* **CollectNewBelief**, **DistributeBelief** *is called in T.*

CommunicateBelief *is associated with the LJF.*

Example 5 Figure 4 shows how belief propagates through a LJF during **CommunicateBelief**. Each node corresponds to an agent in the system. The links between agents corresponds to the communication channel, which may be over a spatial distance. Suppose the operation is initiated at an arbitrarily selected agent represented by T^1. **CollectNewBelief** proceeds by first propagating control from T^1 towards terminal agents along solid arrows, and then propagating belief from terminal agents back to T^1 along dotted arrows. Then **DistributeBelief** proceeds by propagating belief from T^1 towards terminal agents along solid arrows. The time required for control propagation can usually be ignored compared with that for belief propagation.

Note that **CommunicateBelief** is operationally 'semi-parallel'. It is 'parallel' in that T^4 and T^5 may perform **UpdateBelief** relative to their terminal neighbors in parallel during **CollectNewBelief**. It is 'semi'-parallel in that T^4 must perform **UpdateBelief** relative to its terminal neighbors in sequence during **CollectNewBelief**. The same argument can be made on **DistributeBelief** as well.

[1] The operation **CollectBelief** [12] is similar in form to **CollectNewBelief**. But **CollectBelief** deals with a simpler consistency problem and can only be used for initialization. It is thus computationally less expensive than **CollectNewBelief**.

[2] The operation **BeliefInitialization** [12] is similar in form to **CommunicateBelief**. But the former deals with a simpler consistency problem (initialization). It is thus computationally less expensive than **CommunicateBelief**.

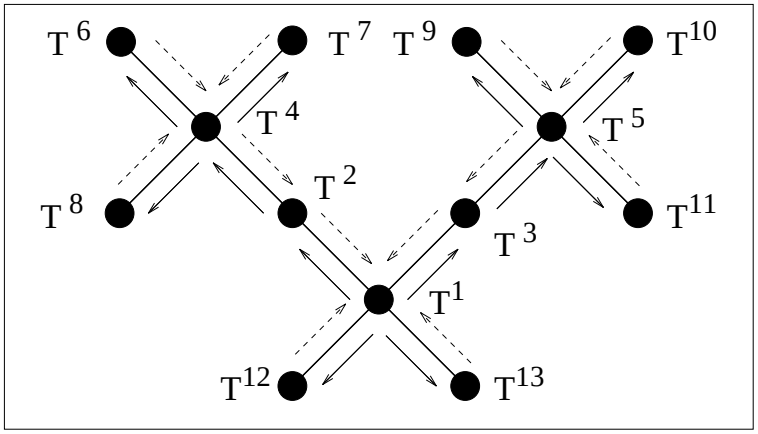

Figure 4: Belief propagation during **CommunicateBelief** in a LJF converted from the MSBN in Figure 1. Each node represents a JT. Each link between two nodes represents the linkages between the JTs, and also corresponds to the communication channel between them. The operation is assumed to be initiated from T^1. Dotted arrows illustrate belief propagation during **CollectNewBelief**, and solid arrows illustrate belief propagation during **DistributeBelief**.

6 Consistency After and Between Communications

We answer the two questions raised in Section 4. First, we show that the operations proposed in Section 5, when performed, guarantee global consistency among multiple agents:

Theorem 6 (Multi-agent consistency) *Let F be a supportive, globally consistent and separable LJF converted from a MSBN of hypertree structure. Let Z be a subset of JTs of F. After the following operations, F is globally consistent.*

1. *For each JT in Z, use* **EnterEvidence** *to enter finite pieces of evidence into the JT.*

2. *Use* **CommunicateBelief** *to communicate belief among JTs in F.*

Proof:

By assumption F is globally consistent before any **EnterEvidence**. We convert F into an imaginary JT Υ and prove the theorem using Υ. For each JT T^i, union all cliques into one huge clique W^i. Equate the belief table (BT) of W^i to $B(T^i)$ (the BT associated with T^i). This is correct since F is globally consistent and separable. For each pair of huge cliques W^i and W^j, equate the BT of their sepset to $B(I^{ij})$ if T^i and T^j are neighbours

in the hypertree structure, where I^{ij} is the interfacing set of variables between T^i and T^j. The resultant graph Υ is a supportive and consistent JT whose joint system belief is proportional to the joint system belief of F since the MSBN has a hypertree structure and F is supportive and globally consistent.

EnterEvidence in T^i of F corresponds to multiplying $B(W^i)$ of Υ by the evidence function. It updates the joint system belief and causes inconsistency. **CommunicateBelief** consists of selection of a JT T^0, and a **CollectNewBelief** followed by a **DistributeBelief**. The **CollectNewBelief** in F corresponds to a **CollectEvidence** in Υ called in W^0. The effect in Υ is that each BT on a sepset becomes the marginalization of the BT of the huge clique most distant from W^0. The **DistributeBelief** in F corresponds to a **DistributeEvidence** in Υ. The resultant Υ is again consistent.

After each **EnterEvidence**, a JT is internally consistent. Both **CollectNewBelief** and **DistributeBelief** call **UpdateBelief** which maintains internal consistency. Therefore, after **CommunicateBelief**, F is locally consistent. The consistency of Υ implies boundary consistency of F. Hence F is globally consistent. □

CommunicateBelief involves computation at each agent and information exchange among multiple agents over 'distance'. Due to the cost involved, **CommunicateBelief** can not be performed frequently. Therefore, each agent may acquire multiple pieces of evidence between two successive **CommunicateBelief** operations, and may have to answer queries before the next **CommunicateBelief** can be performed. The second question we address is: what is the consistency level of these answers to queries. Theorem 7 shows that, between two successive communications, a JT is consistent with all local evidence acquired so far, and is consistent with all global evidence acquired up to the last communication. This is the best that one can expect.

Theorem 7 (Semi-up-to-date) *Let F be a supportive and separable LJF converted from a MSBN of hypertree structure. Let Z be a subset of JTs of F.*

After a **CommunicateBelief** *in F followed by a finite number of* **EnterEvidence** *to each JT in Z, the marginal distributions obtained in a JT $T \in Z$ are identical as would be obtained if only the* **EnterEvidence** *operations in T were performed after the* **CommunicateBelief**.

Proof:

We shall call the **CommunicateBelief** mentioned in the theorem the *first* **CommunicateBelief**. After the operation, F is globally consistent by Theorem 6. Among all the **EnterEvidence** operations performed in Z, only those performed in T change the BTs in T, and none of the other operations has any effect on T. Suppose none of the **EnterEvidence** operations performed outside T was ever performed. We show that if a second **CommunicateBelief** is called in F with T as the initiating JT, the BTs of T will not change at all:

CollectNewBelief called in T does not change BTs in T since F is boundary-consistent after the first **CommunicateBelief**. That **DistributeBelief** does not change

BTs in T is trivially true. □

7 Off-line Time During Communication

7.1 Off-line time of each agent

During **CommunicateBelief**, the BT of a JT T may be changed through **CollectNewBelief** and **DistributeBelief** in the following ways:

1. (Through **CollectNewBelief**) When each of T's neighbour (except the caller of **CollectNewBelief**) has completed its **CollectNewBelief**, the operation **UpdateBelief** that T performs relative to the neighbour may change $B(T)$.

2. (Through **DistributeBelief**) When **DistributeBelief** is called in T, the operation **UpdateBelief** that T performs relative to the caller may change $B(T)$.

It follows from the proof of Theorem 6 that $B(T)$ should not be modified by new external evidence between the first **UpdateBelief** (during **CollectNewBelief**) and the last **UpdateBelief** (during **DistributeBelief**) in the process of **CommunicateBelief**. That is, **EnterEvidence** should not be performed in T between the two **UpdateBelief** operations. Otherwise, **CommunicateBelief** will not regain the global consistency.

Since not being able to process evidence within an interval of time imposes restriction on time-critical applications, the length of the interval should be minimized. We define such interval of time as follows:

Definition 8 (Junction Tree Off-line Time) *Let T be a JT in a LJF F. During a* **CommunicateBelief** *operation in F, let t be the instant of time when the first* **UpdateBelief** *involved by T is started. Let τ be the instant of time when the last* **UpdateBelief** *involved by T is completed. The* **off-line time** *of T during communication is $\Delta(T) = \tau - t$.*

7.2 Factors affecting off-line time

Different JTs in a LJF may have different off-line time during communication depending on several factors:

A **CommunicateBelief** operation is started in a LJF at an arbitrarily selected JT, which we refer to as the communication *root*. The choice of root is one factor that affects the off-line time of each agent in the system.

Example 9 We consider the effect of the root on T^4's off-line time in Figure 4. Since T^1 is selected as the root as shown in the Figure, during **CollectNewBelif**, T^4 must perform **UpdateBelief** relative to T^6, T^7 and T^8 (sequentially) first, and then allow T^2 to perform **UpdateBelief** relative to T^4. Afterwards, T^4 must wait for the completion of

CollectNewBelief in the rest of the system and wait for its turn in **DistributeBelief**. During **DistributeBelief**, T^4 must perform **UpdateBelief** relative to T^2 first and then allow T^6, T^7 and T^8 to perform **UpdateBelief** relative to T^4 (sequentially).

The above order of operations dictates that T^4 becomes off-line as soon as belief propagation from T^6, T^7 and T^8 to T^4 starts. T^4 remains off-line when belief further propagates to T^1 through T^2 and then back to T^4 from T^1. T^4 becomes available after belief propagates back to T^6, T^7, and T^8 through T^4. The total process involves 13 sequential **UpdateBelief** operations in the best case, where T^1 performs **DistributeBelief** relative to T^2 first among its four neighbors (16 in the worst case).

If T^6 instead T^1 is selected as the root, then T^4 will be off-line for only a period of time needed to perform 8 sequential **UpdateBelief**s.

Another factor is the order in which **CollectNewBelief** is performed by each agent relative to its neighbors.

Example 10 Consider T^7 in Figure 4 where the root is T^1. During **CollectNewBelif**, T^4 must perform **UpdateBelief** relative to T^6, T^7 and T^8 sequentially. If T^7 is first selected, T^7 must become off-line before T^6 and T^8, and its off-line time will be prolonged accordingly.

We refer to the order in which multiple neighbors are selected by an agent to perform **UpdateBelief** against, during **CollectNewBelif**, as the *collection order* of the agent.

Similarly, we refer to the order in which multiple neighbors are selected by an agent to perform **UpdateBelief**, during **DistributeBelif**, as the *distribution order* of the agent, which is a third factor affecting each agent's off-line time.

Example 11 Consider T^7 in Figure 4 where the root is T^1. During **DistributeBelief**, T^6, T^7 and T^8 must perform **UpdateBelief** relative to T^4 sequentially. If T^7 is first selected, T^7 can become available before T^6 and T^8, and its off-line time will be shortened accordingly.

The last factor is the time complexity of **UpdateBelief** by a JT T^i relative to a neighbor JT T^k. This time complexity is fixed once the LJF is constructed. Note that the time complexity of **UpdateBelief** by T^i relative to T^k may not be the same as the time complexity of **UpdateBelief** by T^k relative to T^i. This is because **UpdateBelief** performed by T^i relative to T^k involves multiple (the number of linkages between T^i and T^k) performance of **UnifyBelief** in T^i[12]. The time required by **UnifyBelief** in T^i is generally different than that in T^k.

7.3 Off-line time of a multi-agent system

The difference of off-line time across agents calls for a measurement of off-line time of the entire system. We consider two alternatives which may commonly be used:

Definition 12 (Absolute Off-line Time) *Let F be a LJF. Let t be the instant of time when a first JT in F becomes off-line during a* **CommunicateBelief** *operation. Let τ be the instant of time when the last JT becomes available again. The* **absolute off-line time** *of F is $\Delta_{abs} = \tau - t$.*

The absolute off-line time indicates the non-availability of the system as a whole even though some agents are available earlier than others.

Definition 13 (Average Off-line Time) *Let F be a LJF. Let $\Delta(T^i)$ be the off-line time of a JT T^i $(i = 1, \ldots, n)$ in F during a* **CommunicateBelief** *operation. The* **average off-line time** *of F is $\Delta_{ave} = (\sum_{i=1}^{n} \Delta(T^i))/n$.*

The average off-line time indicates the average non-availability of multiple agents.

Both definitions can be modified over a subset of JTs if availability is concerned with only the subset.

7.4 A graphical model for off-line time study

Based on the above analysis, given a LJF and a chosen off-line time measurement, we may manipulate the communication root, collection order and distribution order such that the off-line time is minimized. To avoid distraction by unnecessary details of communication, e.g., the number of linkages and the numerical belief computation, so that we can concentrate on the four factors that determine the off-line time, we abstract communication in a LJF into the following graphical model.

Model 14 (Graphical communication model)

Given an undirected and weighted tree, and an arbitrary node A as the root, the tree is converted to a rooted tree R.

For each node X of R, if $X \neq A$, place an *in-agent* at X. For each node Y of R, if Y has k children, place k *out-agents* at Y.

The agents traverse R according to the following rules:

1. To start with, each parent node Y with leaf children selects one child X, according to some order $O_{in}(Y)$. Once selected, X sends its in-agent to move from X to Y, which takes time $w_{in}(X)$ that is the weight associated with the link (X, Y) in the inward direction (leaf towards root).

   After one child's in-agent arrives at Y, the next child, selected according to $O_{in}(Y)$, sends its in-agent to Y.

   After a parent Y has received all the in-agents from its children, Y is ready for selection by its own parent Z. Once selected by Z, Y sends its in-agent to Z. The inward movement of in-agents continues in this fashion.

2. After the root A receives all in-agents from its children, the inward movement is completed, and an outward movement starts.

   A selects one child X, according to some order $O_{out}(A)$. A then sends one out-agent to move from A to X, which takes time $w_{out}(X)$ that is the weight associated with the link (A, X) in the outward direction.

   After an out-agent of A reaches the destination, A selects another child according to $O_{out}(A)$ and sends another out-agent to the child. The process continues until all out-agents of A reach their destination.

   After an out-agent from A reaches a child X, X selects its own children, according to $O_{out}(X)$, and sends its out-agents to child nodes in sequence.

   The process continues in this fashion until the last out-agent in R reaches its leaf destination, and the outward movement halts.

The above model characterizes the **CommunicateBelief** operation correctly as far as off-line time is concerned:

1. The original undirected tree corresponds to the LJF. Each node corresponds to a JT of the LJF.

2. The root A corresponds to the communication root.

3. The inward movement of in-agents corresponds to belief propagation during **CollectNewBelief**, and the outward movement of out-agents corresponds to belief propagation during **DistributeBelief**.

4. Given a parent node Y and a child node X, $w_{in}(X)$ corresponds to the time required for Y to perform **UpdateBelief** relative to X, and $w_{out}(X)$ corresponds to the time required for X to perform **UpdateBelief** relative to Y.

5. $O_{in}(X)$ corresponds to the collection order (Section 7.2) of X, and $O_{out}(Y)$ corresponds to the distribution order of Y.

6. The time instant when the in-agent of a node X leaves X corresponds to the time instant when the corresponding JT becomes off-line. The time instant when an out-agent from X's parent arrives at X corresponds to the time instant when the corresponding JT becomes available for entering evidence. The interval between the two instants thus corresponds to the off-line time of the JT represented by X.

Note that the graphical model reflects the semi-parallel pattern of communication that is discussed in Example 5. It will be seen that the key aspect of minimization of off-line time is to increase the degree of parallelism by manipulating the relevant factors.

We will use the following notations: We shall say that a non-leaf node X is *off* at the time instant when the in-agent from the first child selected by X starts moving to X. We use $t_{off}(X)$ to denote the instant. If X is a leaf node in the rooted tree, then X is off as soon as its in-agent leaves X.

We shall say that a non-leaf node X is *on* at the time instant when its last out-agent arrives at one of X's children. We use $t_{on}(X)$ to denote the instant. If X is a leaf, then X is on when it receives the out-agent from its parent. We shall say that the off-line time of the node X is $\Delta(X) = t_{on}(X) - t_{off}(X)$.

We shall call the inward movement of in-agents *collection* and the outward movement of out-agents *distribution*. For collection, we use $t_{rdy}(Y)$ to denote the time instant when a non-leaf node Y receives the last in-agent from its children and is *ready* for its parent to select. For a leaf node Y, we assign $t_{rdy}(Y)$ to be the instant when collection starts. We use $t_{wat}(Y)$ to denote the time instant when Y's in-agent arrives at its parent and Y starts to *wait* for an out-agent to be sent from its parent. For the root A, we assign $t_{wat}(A) = t_{rdy}(A)$.

For distribution, we use $t_{sel}(X)$ to denote the time instant when a node X is *selected* by its parent Y such that Y is about to send an out-agent to X. For the root A, we assign $t_{sel}(A)$ to be the instant when distribution starts. We use $t_{cpt}(X)$ to denote the time instant when X receives the out-agent from Y (distribution relative to Y is *completed*). For the root A, we assign $t_{cpt}(A) = t_{sel}(A)$.

Example 15 Figure 5 illustrates the graphical communication model for a seven-agent system. The figure in the left shows the rooted tree R with the root A, and leaves D, E, F and G. It also shows the in-agent and out-agents of each node, and the in-weight and the out-weight of each link.

The figure in the middle shows collection in the rooted tree R. The collection order used is from left to right for each parent node. At time instant $t = 0$, E is off. Its in-agent arrives at B at $t = 5$. Thus E starts to wait for distribution at $t = 5$, and B is ready for selection by A at the same point in time.

Parallel to the above activity, at $t = 0$, F (the leftmost child) is selected by C to send its in-agent which arrives at C at $t = 1$. Then C selects the next child G, and G's in-agent arrives at C at $t = 4$. Thus, F is waiting at $t = 1$, G is waiting at $t = 4$, and C is ready at $t = 4$.

According to the left-to-right order, A selects B at $t = 5$. B's in-agent arrives at A at $t = 9$. Then C is selected, whose in-agent arrives at A at $t = 11$. Then D is selected, whose in-agent arrives at $t = 19$, and collection is completed.

The figure in the right shows distribution which follows collection immediately. The distribution order used is right-to-left. At $t = 19$, A sends an out-agent to D, which arrives at $t = 26$. D is on at this moment since it has no child.

C is then selected and another out-agent of A arrives at C at $t = 29$.

The last out-agent of A arrives at B at $t = 35$, and A is on at this moment. Parallel to the movement of the last out-agent of A, at $t = 29$, C sends its first out-agent to G

which arrives at $t = 31$, and then G is on. C then sends another out-agent to F which arrives at $t = 35$. At this moment, both C and F are on.

At $t = 35$, B sends its out-agent to E, which arrives at $t = 40$. At this moment, both B and E are on. Distribution, as well as the entire communication, is then completed.

All activities during the communication are full specified by the tuples used to label nodes in figures in the middle and in the right. The off-line time of each node and the entire system can thus be calculated. For instance, $\Delta(C) = 35 - 0 = 35$ and $\Delta(D) = 26 - 11 = 15$. The absolute off-line time of the system is $\Delta_{abs} = 40 - 0 = 40$. The average off-line time is $\Delta_{ave} = (30 + 40 + 35 + 15 + 40 + 35 + 30)/7 = 32.1$.

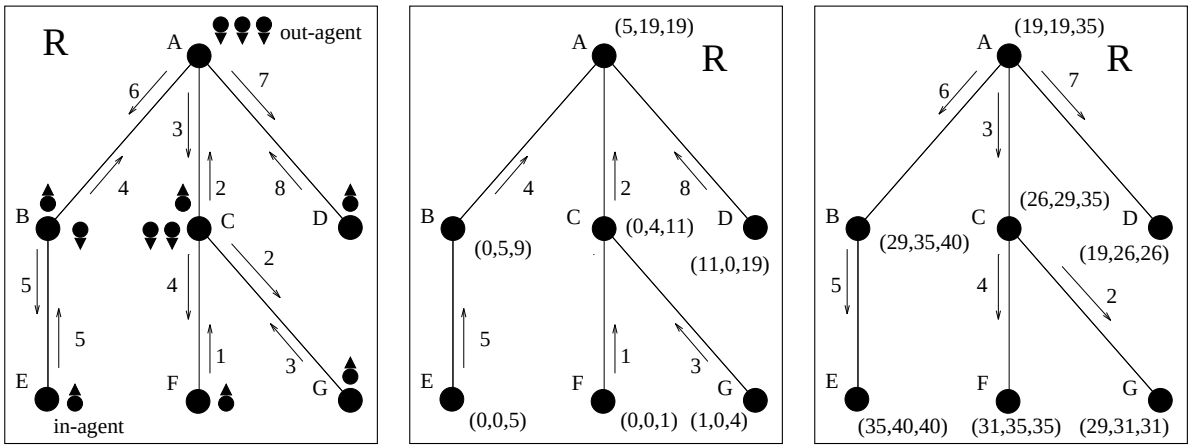

Figure 5: A graphical model for communication in a seven-agent system. The in-weight of a link is indicated by an upward arrow and the associated label, and the out-weight of a link is indicated by a downward arrow and the associated label. Left: The weighted tree R rooted at A. The in-agent and out-agents of each node, as well as the in-weight and the out-weight of each link are shown. Middle: Collection in R with only the in-weight of each link shown. Each node X is labeled will a triple $(t_{off}(X), t_{rdy}(X), t_{wat}(X))$. Right: Distribution in R with only the out-weight of each link shown. Each node X is labeled will a triple $(t_{sel}(X), t_{cpt}(X), t_{on}(X))$.

In the middle (right) figure of Example 15, we outlined the timing of all the activities involved in collection (distribution). We will refer to such a specification of timing of every node's activity during collection (distribution) as a collection (distribution) *schedule*.

Our goal is to find schedules with minimal off-line time. In Example 15, we assumed, in both schedules, that a node engages in its activity as soon as the activity is possible without any delay. Since unnecessary idling can not contribute positively to our goal, we will exclude from our consideration those schedules in which some nodes delay their activities unnecessarily. On the other hand, if there is any practical reason to delay the

belief propagation, e.g., computer network delay, we assume that the delay has been modeled in the link weights.

The schedule in Example 15 is not optimal. We illustrate this using absolute off-line time: During collection, A follows the left-to-right order, and thus waits until $t = 5$ when B is ready. However, another child D of A is ready at $t = 0$, but is selected only at $t = 11$. If D is selected first, both A and D can be engaged in their activity earlier, and A will complete collection and start distribution earlier. The consequence is a shorter absolute off-line time. The difference made by switching the first child to D is that the resultant schedule has a higher degree of parallelism: The parallel activities E to B, and F, G to C in the previous schedule are now also paralleled by the activity D to A.

In the remaining part of this paper, we use the graphical model to study minimization of off-line time with an arbitrarily given communication root. Limited by the space, the optimization of the root is beyond the scope of this paper.

8 Communication Schedules with Minimal Absolute Off-line Time

This section derives the collection and distribution schedules that provide the minimal absolute off-line time, given a rooted tree.

Let collection start at time instant $t = t_0$ and terminate at $t = t_1$. Let distribution start at time instant $t = t_1$ and terminate at $t = t_2$. Denote the interval of time between t_0 and t_1 by Δ_{0-1}, and denote the interval of time between t_1 and t_2 by Δ_{1-2}. We have $\Delta_{abs} = \Delta_{0-1} + \Delta_{1-2}$. Since Δ_{0-1} is independent of Δ_{1-2},

$$min(\Delta_{abs}) = min(\Delta_{0-1}) + min(\Delta_{1-2})$$

where the minimization in the left-hand side of the equation is over all collection and distribution schedules, the first minimization in the right-hand side is over all collection schedules, and the second in the right-hand side is over all possible distribution schedules. We can thus study the optimal collection schedules and the optimal distribution schedules separately.

8.1 Optimal collection schedules

Collection starts from leaves of the rooted tree. Consider first a parent node Y with leaf children $X_1, \ldots, X_n$. The earliest instant of time when Y is ready is $min(t_{rdy}(Y)) = \sum_{i=1}^{n} w_{in}(X_i)$, where minimization is over all collection orders of Y. The order $O_{in}(Y)$ has no effect on $min(t_{rdy}(Y))$. Thus as far as $min(\Delta_{0-1})$ is concerned, the n children of Y can be treated as a single child X' with $w_{in}(X') = \sum_{i=1}^{n} w_{in}(X_i)$.

For example, in Figure 5, $min(t_{rdy}(C)) = 1 + 3$ no matter F is first selected or G is first selected. Replacing links (C, F) and (C, G) by a link (C, X') with $w_{in}(X') = 4$ does not affect the value of $min(t_{rdy}(C))$.

Let us further consider a node Y with children $X_1, \ldots, X_n$ each of which is possibly an internal node. Suppose we have minimized $t_{rdy}(X_i)$ $(i = 1, \ldots, n)$ (if X_i is a leaf, $t_{rdy}(X_i) = t_0$). We compare $t_{rdy}(Y)$ resultant from different collection orders to find one that minimizes $t_{rdy}(Y)$. We use an ordered tuple to denote a particular order. For example, $O_{in}(Y) = (X_1, X_2, \ldots, X_n)$ denotes the order of collection starting from X_1, followed by X_2, and so on.

We start with $n = 2$ (Y has only two children). There are only two possible orders: $O_{in}^{(1)}(Y) = (X_1, X_2)$ and $O_{in}^{(2)}(Y) = (X_2, X_1)$. Suppose $t_{rdy}(X_1) \leq t_{rdy}(X_2)$. We show that $t_{rdy}(Y)$ is minimized with $O_{in}^{(1)}(Y)$:

We consider two exclusive and exhaustive cases of timing: $t_{rdy}(X_1) + w_{in}(X_1) \leq t_{rdy}(X_2)$ and $t_{rdy}(X_1) + w_{in}(X_1) > t_{rdy}(X_2)$.

When $t_{rdy}(X_1) + w_{in}(X_1) \leq t_{rdy}(X_2)$, if $O_{in}^{(1)}(Y)$ is followed, then $t_{rdy}^{(1)}(Y) = t_{rdy}(X_2) + w_{in}(X_2)$. If $O_{in}^{(2)}(Y)$ is followed, then $t_{rdy}^{(2)}(Y) = t_{rdy}(X_2) + w_{in}(X_2) + w_{in}(X_1)$. We have $t_{rdy}^{(1)}(Y) < t_{rdy}^{(2)}(Y)$.

On the other hand, when $t_{rdy}(X_1) + w_{in}(X_1) > t_{rdy}(X_2)$, if $O_{in}^{(1)}(Y)$ is followed, then $t_{rdy}^{(1)}(Y) = t_{rdy}(X_1) + w_{in}(X_1) + w_{in}(X_2)$. If the order $O_{in}^{(2)}(Y)$ is followed, then $t_{rdy}^{(2)}(Y) = t_{rdy}(X_2) + w_{in}(X_2) + w_{in}(X_1)$. Again, we have $t_{rdy}^{(1)}(Y) \leq t_{rdy}^{(2)}(Y)$. Therefore, when $t_{rdy}(X_1) \leq t_{rdy}(X_2)$, the order (X_1, X_2) always minimizes $t_{rdy}(Y)$.

We can generalize the above result with induction. Let the parent Y have $n = k \geq 2$ children $X_1, \ldots, X_n$ with $t_{rdy}(X_1) \leq t_{rdy}(X_2) \leq \ldots \leq t_{rdy}(X_n)$. Assume $t_{rdy}(Y)$ is minimized with a collection order $O_{in}(Y) = (X_1, \ldots, X_n)$ when $n = k$. Denote the minimal $t_{rdy}(Y)$ by τ.

Suppose we add one more child X_{k+1} to Y such that $t_{rdy}(X_k) \leq t_{rdy}(X_{k+1})$. Denote the minimal $t_{rdy}(Y)$ in the case of $k + 1$ children by ρ. We consider two exclusive and exhaustive cases of timing: $\tau < t_{rdy}(X_{k+1})$ and $\tau \geq t_{rdy}(X_{k+1})$.

By the inductive assumption, the completion time of collection at $X_1, \ldots, X_k$ is minimized to τ with the order $O_{in}(Y)$. If $\tau < t_{rdy}(X_{k+1})$, collection at X_{k+1} can not possibly start before τ. Therefore, $\rho = t_{rdy}(X_{k+1}) + w_{in}(X_{k+1})$ when collection order is $O_{in}^{(1)}(Y) = (X_1, \ldots, X_k, X_{k+1})$.

If $\tau \geq t_{rdy}(X_{k+1})$, collection at X_{k+1} may start before τ. If we follow $O_{in}^{(1)}(Y)$, we obtain $t_{rdy}^{(1)}(Y) = \tau + w_{in}(X_{k+1})$. This ready time in the case of $k + 1$ children can not be improved. If it could be, since collection at any child X_i $(i \in \{1, \ldots, k\})$ can be performed before collection at X_{k+1}, it implies that collection at $X_1, \ldots, X_k$ can also be improved to be less than τ, which is contradictory to the inductive assumption that $O_{in}(Y)$ is optimal. Therefore, $O_{in}^{(1)}(Y)$ is optimal in the case $n = k + 1$.

The above statement about the optimal collection order is thus established.

What is the $t_{rdy}(Y)$ for a node Y under a particular order? Let the order be $O_{in}(Y) = (X_1, \ldots, X_n)$. The instant when collection with X_1 completes is $t_{wat}(X_1) = t_{rdy}(X_1) + w_{in}(X_1)$. The instant when collection with X_2 completes is $t_{wat}(X_2) = max(t_{rdy}(X_1) + w_{in}(X_1), t_{rdy}(X_2)) + w_{in}(X_2) = max(t_{rdy}(X_1) + w_{in}(X_1) + w_{in}(X_2), t_{rdy}(X_2) + w_{in}(X_2))$.

Generalizing the formula to X_n, we obtain

$$t_{rdy}(Y) = max(t_{rdy}(X_1) + \sum_{i=1}^{n} w_{in}(X_i), t_{rdy}(X_2) + \sum_{i=2}^{n} w_{in}(X_i), \ldots, t_{rdy}(X_n) + \sum_{i=n}^{n} w_{in}(X_i)).$$

We summerize the above analysis in Algorithm 16, Theorem 18, and Corollary 19. Algorithm 16 arranges the left-right order of children for each node such that the preferred collection order becomes topologically explicit.

Algorithm 16 (Order arrangement for collection)

Input: *A rooted tree of depth M with the in-weight of each link defined. (The depth of root is 0.)*

Output: *The rooted tree with the left-right order of children of each parent re-arranged.*

begin

D := M-1

for each leaf Z of depth D, do

$t_{rdy}(Z) := 0$

for each node Y of depth D with child nodes $X_1, \ldots, X_n$, do

$t_{rdy}(Y) := \sum_{i=1}^{n} w_{in}(X_i)$

D := D-1

while $D \geq 0$, do

for each node Y of depth D with n child nodes, do

arrange children of Y and index them from left to right as $X_1, \ldots, X_n$ such that $t_{rdy}(X_1) \leq \ldots \leq t_{rdy}(X_n)$

$t_{rdy}(Y) := max(t_{rdy}(X_1) + \sum_{i=1}^{n} w_{in}(X_i), \ldots, t_{rdy}(X_n) + \sum_{i=n}^{n} w_{in}(X_i))$

for each leaf Z of depth D, do

$t_{rdy}(Z) := 0$

D := D-1

end

Example 17 Figure 6 illustrates Algorithm 16. The depth of the tree is $M = 2$. After the second *for* loop, $t_{rdy}(B) = 5$, $t_{rdy}(C) = 4$, and $t_{rdy}(D) = 0$. The *while* loop is processed only once. After the loop, the children of A is arranged in the order D, C, B from left to right, and $t_{rdy}(A) = max(0 + 8 + 2 + 4, 4 + 2 + 4, 5 + 4) = 14$.

Theorem 18 establishes that, for each node, the interval of time between t_0 and completion of collection at the node, relative to its children, is minimal if the proposed order is followed.

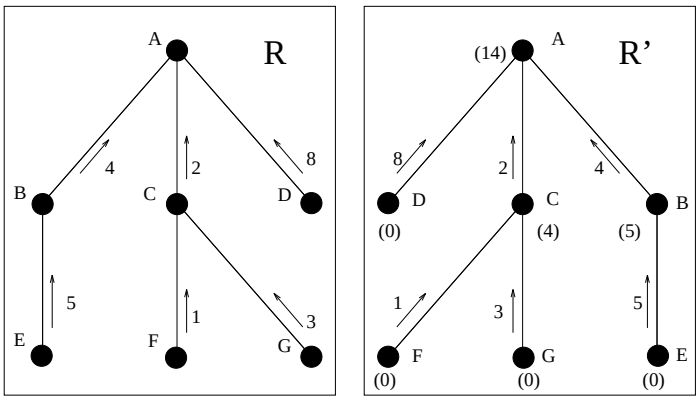

Figure 6: R: A rooted tree for collection. R': R after processed according to Algorithm 16. Each node X of R' is labeled with $(t_{rdy}(X))$ as computed by Algorithm 16.

Theorem 18 (Optimal collection order)

Let R be a rooted tree for collection.

For each parent node Y, the instant of time $t_{rdy}(Y)$ is minimized if R is arranged according to Algorithm 16 and collection at each node is performed according to the left-to-right order.

The minimal value of $t_{rdy}(Y)$ is as computed by Algorithm 16.

Let the node Y in Theorem 18 be the root of R, we obtain the following Corollary 19 which establishes the optimal collection schedule relative to absolute off-line time.

Corollary 19 (Optimal collection schedule)

Let R be a tree for collection rooted at A.

The absolute off-line time Δ_{0-1} is minimized if R is arranged according to Algorithm 16 and then collection at each node is performed according to the left-to-right order.

The minimal value of Δ_{0-1} is given by $t_{rdy}(A)$ as computed by Algorithm 16.

Example 20 The minimal absolute off-line time $min(\Delta_{0-1})$ for R in Figure 6 is obtained using R' and the left-to-right collection order: $min(\Delta_{0-1}) = t_{rdy}(A) = 14$. It is a 26% improvement over $t_{rdy}(A) = 19$ in Example 15.

Note that the above optimal order is essentially a first-come-first-serve policy. The minimal absolute off-line time should be reached when the maximal degree of parallelism is achieved during collection. It makes sense that a first-come-first-serve policy will provide such parallelism.

In Example 20, the same minimal Δ_{0-1} can be obtained if the root performs collection in the order D, B, C, instead of D, C, B. Given a rooted tree, optimal collection (distribution) schedules are generally not unique.

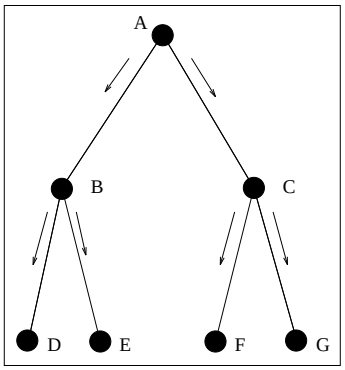

Figure 7: A rooted tree of depth 2 for distribution.

8.2 Optimal distribution schedules

Distribution starts from the root. Consider first a rooted tree with depth 2 and with branching factor 2 as shown in Figure 7. Distribution in the root A can be performed with two possible orders: $O^{(1)}_{out}(A) = (B, C)$ and $O^{(2)}_{out}(A) = (C, B)$.

With $O^{(1)}_{out}(A)$,

$$\Delta^{(1)}_{1-2} = max(w_{out}(B) + w_{out}(C) + w_{out}(F) + w_{out}(G), w_{out}(B) + w_{out}(D) + w_{out}(E)).$$

Since distribution from a parent of leaves to the leaf children is sequential, all leaf children of the same parent can be treated equivalently as a single leaf with its w_{out} being the sum of w_{out}s of the original leaves involved. We can therefore write

$$\Delta^{(1)}_{1-2} = max(w_{out}(B) + w_{out}(C) + w_{out}(FG), w_{out}(B) + w_{out}(DE)),$$

where $w_{out}(FG) = w_{out}(F) + w_{out}(G)$ and FG denotes the equivalent single leaf.

Similarly, with $O^{(2)}_{out}(A)$,

$$\Delta^{(2)}_{1-2} = max(w_{out}(C) + w_{out}(B) + w_{out}(DE), w_{out}(C) + w_{out}(FG)).$$

The optimal order is the one with a smaller Δ_{1-2}.

We claim that if $w_{out}(DE) \leq w_{out}(FG)$, then $O^{(2)}_{out}(A)$ is the optimal order: Under the condition, $\Delta^{(1)}_{1-2}$ can be simplified into $\Delta^{(1)}_{1-2} = w_{out}(B) + w_{out}(C) + w_{out}(FG)$ which is larger than or equal to both entries of $\Delta^{(2)}_{1-2}$, namely, $w_{out}(C) + w_{out}(B) + w_{out}(DE)$ and $w_{out}(C) + w_{out}(FG)$.

The above result suggests that the out-weights at the top level (i.e., $w_{out}(B)$ and $w_{out}(C)$) are not critical, but the sums of out-weights at the bottom level (i.e., $w_{out}(DE)$ and $w_{out}(FG)$) are. This result in fact is general as is shown in Proposition 21.

Proposition 21 (Optimal distribution schedule in trees of depth 2)

Let R be a tree rooted at A for distribution. Let the depth of R be 2. Let the children of root be $X_1, \ldots, X_n$. Let the sum of out-weights of children of X_i be v_i such that $v_1 \leq v_2 \leq \ldots \leq v_n$.

The absolute distribution off-line time Δ_{1-2} in R is minimized if the distribution order of A is $O_{out}(A) = (X_n, \ldots, X_1)$.

Proof:

According to the order $O^{(1)}_{out}(A) = (X_n, \ldots, X_1)$, the distribution time is

$$\begin{aligned}\Delta^{(1)}_{1-2} = max(\quad & w(X_n) + v_n, \\ & w(X_n) + w(X_{n-1}) + v_{n-1}, \\ & \ldots, \\ & w(X_n) + \ldots + w(X_i) + v_i, \\ & \ldots, \\ & w(X_n) + \ldots + w(X_1) + v_1)\end{aligned}$$

where we have written $w(X_i)$ instead of $w_{out}(X_i)$ for simplicity. Note we have listed the entries in $max()$ in the order consistent with the order in which X_i appears in $O^{(1)}_{out}(A)$. In this order, if $w(X_i)$ appears in an entry, it appears in every entry to its right. Note also that each entry has exactly one v among its addends.

Let $O^{(2)}_{out}(A) = (X'_n, \ldots, X'_1)$ be any order distinct from $O^{(1)}_{out}(A)$ where X'_i is some X_k ($k \in \{1 \ldots n\}$) and $X'_i \neq X'_j$ for $i \neq j$. We denote the weight associated with X'_i by $w(X'_i)$, and the sum of weights of children of X'_i by v'_i. The distribution time under $O^{(2)}_{out}(A)$ is

$$\Delta^{(2)}_{1-2} = max(e'_n, e'_{n-1}, \ldots, e'_1) \text{ where } e'_i = (\sum_{k=i}^{n} w(X'_k)) + v'_i.$$

First we simplify $\Delta^{(2)}_{1-2}$ by removing superfluous entries. Suppose v_n is an addend of e'_d and $d < n$, or equivalently, X_n is not the first in $O^{(2)}_{out}(A)$. Then $e'_n, \ldots, e'_{d+1}$ can be eliminated from $max()$ without altering the value of $\Delta^{(2)}_{1-2}$, since $v_n = max^n_{i=1}(v_i)$ and therefore $e'_i < e'_d$ when $i > d$. After the elimination, we have $\Delta^{(2)}_{1-2} = max(e'_d, e'_{d-1}, \ldots, e'_1)$.

Suppose now v_m ($m \in \{1 \ldots n-1\}$) is an addend of e'_c, and m is the highest index value among vs contained in e'_{d-1} through e'_1. With the same argument as above, $e'_{d-1}, \ldots, e'_{c+1}$ can all be eliminated from $max()$.

Repeating this process, we end up with $\Delta^{(2)}_{1-2} = max(e'_d, e'_c, \ldots, e'_a)$ where v_n is an addend of e'_d, v_m is an addend of e'_c, $\ldots$, v_l is an addend of e'_a, such that $n > m > \ldots > l$. We will refer to the expression of $\Delta^{(2)}_{1-2}$ before simplification as the *expanded* form, and refer to the expression after simplification as the *simplified* form.

(Example) suppose $O^{(2)}_{out}(A) = (X_4, X_5, X_2, X_3, X_1)$. The expanded form of $\Delta^{(2)}_{1-2}$ is

$$\begin{aligned}\Delta^{(2)}_{1-2} &= max(w(X_4)+v_4, w(X_4)+w(X_5)+v_5, w(X_4)+w(X_5)+w(X_2)+v_2,\\ &\quad w(X_4)+w(X_5)+w(X_2)+w(X_3)+v_3,\\ &\quad w(X_4)+w(X_5)+w(X_2)+w(X_3)+w(X_1)+v_1).\end{aligned}$$

After removing superfluous entries, the simplified form is

$$\begin{aligned}\Delta^{(2)}_{1-2} &= max(w(X_4)+w(X_5)+v_5, w(X_4)+w(X_5)+w(X_2)+w(X_3)+v_3,\\ &\quad w(X_4)+w(X_5)+w(X_2)+w(X_3)+w(X_1)+v_1).\end{aligned}$$

Next, we prove a lemma which states the following:

(Lemma) After $\Delta^{(2)}_{1-2}$ is simplified into $\Delta^{(2)}_{1-2} = max(e'_d, \ldots, e'_c, \ldots, e'_a)$, if $e'_c = (\sum w(X_i)) + w(X_m) + v_m$, then all of $w(X_n), \ldots, w(X_{m+1})$ are addends in $\sum w(X_i)$.

It suffices to show that e'_c has all of $w(X_n), \ldots, w(X_{m+1})$ as its addends.

Consider v_i $(n \geq i \geq m+1)$. In the expanded form of $\Delta^{(2)}_{1-2}$, v_i appears as an addend in an entry that is either before (to the left of) e'_c or after e'_c. In the following, we simply say that v_i appears before (after) e'_c.

If v_i appears before e'_c, then e'_c must contain the addend $w(X_i)$. It can not appear after e'_c, since entries in the simplified $\Delta^{(2)}_{1-2}$ has a decreasing order for the index of addend v. If v_i had appeared after e'_c in the expanded form of $\Delta^{(2)}_{1-2}$, then e'_c would have been eliminated in the simplification process. The lemma is then proven.

Finally, we show that $\Delta^{(1)}_{1-2} \leq \Delta^{(2)}_{1-2}$ by identifying one entry e'_c in the simplified $\Delta^{(2)}_{1-2}$ such that $\Delta^{(1)}_{1-2} \leq e'_c$.

Suppose $\Delta^{(1)}_{1-2} = (\sum_{i=k}^{n} w(X_i)) + v_k$ $(k \in \{1, \ldots, n\})$. We search for an entry e'_c in the simplified $\Delta^{(2)}_{1-2}$ such that e'_c has v_k as an addend. We consider the following three exclusive and exhaustive cases:

(Case 1) If such an e'_c is found, by the above lemma, we have $\Delta^{(1)}_{1-2} \leq e'_c$.

If an entry with the addend v_k is not found, we search for an entry e'_c with an entry e'_b next to its right such that e'_c has an addend v_l $(l > k)$ and e'_b has an addend v_j $(k > j)$.

(Case 2) If such e'_c and e'_b are found, we show that e'_c has all of $w(X_n), \ldots, w(X_k)$ as its addends.

By the lemma above, e'_c has all of $w(X_n), \ldots, w(X_l)$ as its addends. We need to show that e'_c also has all of $w(X_{l-1}), \ldots, w(X_k)$ as its addends.

By the lemma, e'_b must have all of $w(X_{l-1}), \ldots, w(X_k)$ as its addends. Therefore, in the expanded form of $\Delta^{(2)}_{1-2}$, all of $v_{l-1}, \ldots, v_k$ must have appeared as addends in entries to the left of e'_b. The question is whether they appear before e'_c or after e'_c.

If any v_i ($i \in \{l-1, \ldots, k\}$) had appeared in an entry after e'_c in the expanded form of $\Delta^{(2)}_{1-2}$, since $i > j$, at least one such entry appeared would have been included in the simplified $\Delta^{(2)}_{1-2}$. Therefore, each v_i ($i \in \{l-1, \ldots, k\}$) must have appeared before e'_c, and e'_c has $w(X_i)$ for $i = l-1, \ldots, k$ as addends.

(Case 3) If the above specified e'_c is found but there exists no e'_b as specified above, i.e., e'_c is the right-most entry in the simplified form of $\Delta^{(2)}_{1-2}$, we show that e'_c contains addends $w(X_n), \ldots, w(X_k)$.

Consider the left-most entry e'_1 in the expanded form of $\Delta^{(2)}_{1-2}$. This entry contains all of $w(X_n), \ldots, w(X_1)$. The simplification process does not eliminate the right-most entry of the expanded $\Delta^{(2)}_{1-2}$. Therefore, the simplified form has the identical right-most entry as the expanded form: $e'_c = e'_1$ which contains addends $w(X_n), \ldots, w(X_k)$.

We have shown that $\Delta^{(1)}_{1-2} \leq \Delta^{(2)}_{1-2}$ holds for an arbitrary order $O^{(2)}_{out}(A)$. Therefore, $O^{(1)}_{out}(A)$ minimizes Δ_{1-2}. □

Example 22 According to Proposition 21, Δ_{1-2} for R in Figure 5 will be minimized if $O_{out}(A) = (C, B, D)$. The minimal Δ_{1-2} is $\Delta_{1-2} = max(3+(4+2), 3+6+5, 3+6+7+0) = 16$ which is a 24% improvement over $\Delta_{1-2} = 21$ in Example 15. A corresponding optimal schedule is shown in Figure 8. The distribution order used is left-to-right.

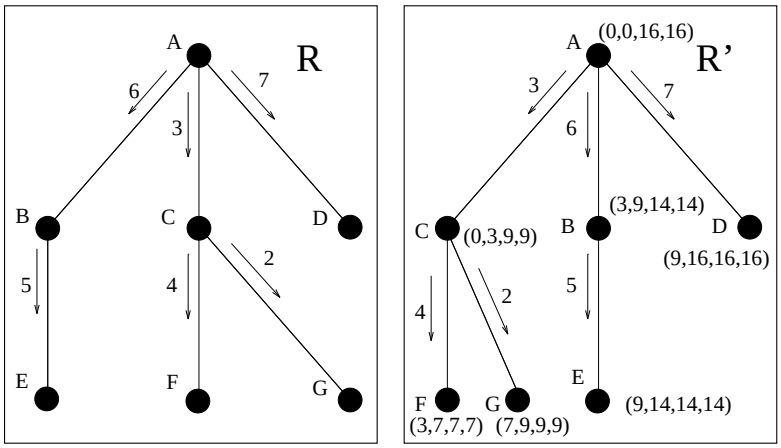

Figure 8: R: A rooted tree for distribution. R': R with the left-right order of nodes B, C, D rearranged according to $O_{out}(A) = (C, B, D)$ determined by Proposition 21. The distribution schedule is shown by the label $(t_{sel}(X), t_{cpt}(X), t_{on}(X), t_{on}(X) - t_d)$ at each node X. The distribution starts at $t_d = 0$.

Algorithm 23 and Theorem 25 generalize Proposition 21 to an arbitrary rooted tree for distribution. Algorithm 23 rearranges the left-right order of children for each node such

that the optimal distribution order becomes topologically explicit. Theorem 25 establishes the optimal schedule.

Algorithm 23 (Order arrangement for distribution)
Input: A rooted tree of depth M with the out-weight of each link defined.
Output: The rooted tree with the left-right order of children of each parent re-arranged.
begin
D := M-1
for each leaf Z of depth D, do
$v(Z) := 0$
for each node Y of depth D with child nodes $X_1, \ldots, X_n$, do
$v(Y) := \sum_{i=1}^{n} w_{out}(X_i)$
D := D-1
while $D \geq 0$, do
for each node Y of depth D with n child nodes, do
arrange children of Y and index them from left to right as $X_1, \ldots, X_n$ such that $v(X_1) \leq \ldots \leq v(X_n)$
$v(Y) := max(e_n, \ldots, e_1)$ *where* $e_i = (\sum_{k=i}^{n} w_{out}(X_k)) + v(X_i)$
for each leaf Z of depth D, do
$v(Z) := 0$
D := D-1
end

Example 24 Figure 9 illustrates Algorithm 23. After the first two *for* loops, we have $v(E) = 8$, $v(F) = 14$, $v(G) = 0$, $v(H) = 15$, $v(I) = 0$, and $v(J) = 18$. After the first pass of the *while* loop, children of B are arranged from left to right in the order G, E, F based on their v values. Children of D are arranged from left to right in the order I, H, J. The v values for nodes B, C, D are assigned as $v(B) = 19$, $v(C) = 0$, $v(D) = 27$.

After the second pass of the *while* loop, children of root A are arranged from left to right in the order C, B, D, and $v(A)$ is assigned the value $v(A) = 36$.

Theorem 25 (Optimal distribution schedule)
Let R be a rooted tree for distribution with root A. The absolute off-line time Δ_{1-2} is minimized if R is arranged according to Algorithm 23 and the distribution order for each node is right-to-left.

The minimal Δ_{1-2} is given by $v(A)$ as computed by Algorithm 23.

Proof:

We prove by induction. Let the depth of R be M. The statement is true if $M = 2$ according to Proposition 21.

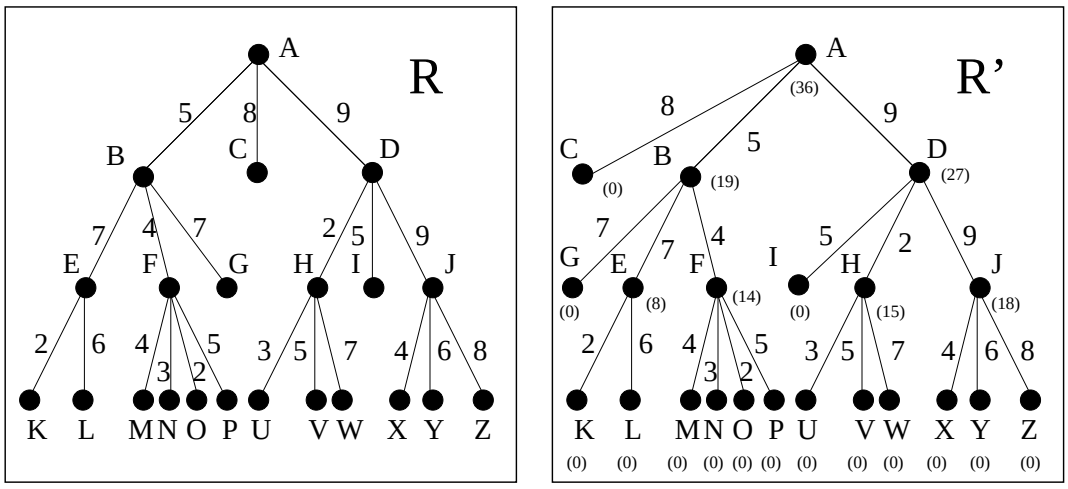

Figure 9: R: A rooted tree for distribution operation. R': R after being processed according to Algorithm 23. Each node X is labeled with $(v(X))$ as computed by Algorithm 23.

Assume that the statement is true for $M = K$ $(K \geq 2)$. Consider $M = K+1$. Assume that R is arranged according to Algorithm 23, and the root A of R has n $(n \geq 1)$ children $Y_1, \ldots, Y_n$. Each child is the root of a subtree.

By the inductive assumption, the distribution time for the subtree rooted at Y_i $(i \in \{1, \ldots, n\})$ is minimized if the right-to-left order is followed. The minimal time is $v(Y_i)$.

For $i = 1, \ldots, n$, replace the subtree rooted at Y_i by a single link (Y_i, X_i) with $w_{out}(X_i) = v(Y_i)$. The resultant is a tree of depth 2 rooted at A that satisfies the condition of Proposition 21. Therefore distribution using the right-to-left order at A will optimize the distribution time. Since distribution time for each subtree rooted at Y_i is minimized by induction assumption, and the distribution at root A is also optimized, Δ_{1-2} is minimal for $M = k + 1$ and the minimal value is given as $v(A)$. $\square$

Example 26 The absolute distribution off-line time for R in Figure 9 is minimized by arranging R as R' and following the right-to-left distribution order. The minimal Δ_{1-2} is $v(A) = 36$.

9 Duality of Collection and Distribution

So far, we have developed independently collection and distribution schedules with minimal absolute off-line time. However, if we compare the two activities as well as the resultant optimal schedules, great similarity can be uncovered.

First of all, the two activities proceed through the tree structure in a similar semi-parallel pattern. Inward movement of in-agents from the children of a node must proceed

sequentially. So must outward movement of out-agents of a node to its children. Two nodes of a common ancestor may be engaged in collection relative to their children respectively in parallel. So may they be engaged in distribution relative to their children in parallel.

Second, let us compare the Algorithm 16 and Algorithm 23 which arrange the order of child nodes for each parent such that minimal absolute off-line time can be obtained for collection and distribution, respectively. If we replace w_{in} in Algorithm 16 by w_{out}, it would arrange the nodes of a tree in exactly the same way as Algorithm 23. Comparing the optimal collection schedule in Corollary 19 and the optimal distribution schedule in Theorem 25, we see that one follows the left-to-right order for each node and the other follows the right-to-left order. Given that the two operations direct information to the opposite directions, and the tree is arranged in the same way, it makes sense for the two operations to have the opposite orders.

We establish the duality of the two activities relative to off-line time in Theorem 27.

Theorem 27 (Duality)

Let R be a weighted tree rooted at A with the identical in-weight and out-weight at each link.

Let S_d be a distribution schedule which starts from A at t_d and terminates at τ_d. Let the distribution order of a parent node Y be denoted as $O_d(Y)$ in this schedule.

A schedule S_c for collection, that starts at t_c and terminates at A at τ_c, can be obtained, by requiring each parent node Y to follow a collection order that is opposite to $O_d(Y)$, such that the followings hold:

1. *$\tau_c - t_c = \tau_d - t_d$, and*
2. *for every node X, $\tau_c - t_{off}(X)$ in S_c is identical to $t_{on}(X) - t_d$ in S_d.*

The converse (starting from S_c to obtain S_d) is also true.

Proof:

We start with a S_d and construct a S_c such that the above two conditions are satisfied. Once the mapping is created, the converse of the theorem is trivially true.

We assume that for each parent node Y with m child nodes, the child nodes are arbitrarily indexed as $X_1, \ldots, X_m$. Without losing generality, we denote the distribution order of Y in S_d by $O_d(Y) = (X_1, \ldots, X_m)$. Using the notation defined in Section 7.4, we can then characterize S_d as follows:

$$\begin{cases} t_{cpt}(Y) = t_d & \text{if } Y \text{ is the root} \\ t_{cpt}(Y) = t_{sel}(Y) + w(Y) & \text{if } Y \text{ is not the root} \\ t_{on}(Y) = t_{cpt}(Y) + \sum_{k=1}^{m} w(X_k) & \text{if } Y \text{ is not a leaf} \\ t_{sel}(X_i) = t_{cpt}(Y) + \sum_{k=1}^{i-1} w(X_k) & \text{if } X_i \text{ is not the root} \\ t_{on}(X) = t_{cpt}(X) & \text{if } X \text{ is a leaf} \\ \tau_d = max(t_{on}(Z)) & max() \text{ over every node } Z \end{cases} \quad (1)$$

We construct a collection schedule S_c from S_d by requiring each parent node Y to follow the collection order $O_c(Y) = (X_m, \ldots, X_1)$ which is opposite to $O_d(Y)$. The resultant schedule S_c can be characterized as follows:

$$\begin{cases} t_{off}(X) = \tau_d - t_{on}(X) + t_c & \text{if } X \text{ is a leaf} \\ t_{off}(Y) = t_{off}(X_m) & \text{if } Y \text{ is not a leaf} \\ t_{rdy}(X) = t_c & \text{if } X \text{ is a leaf} \\ t_{rdy}(Y) = t_{off}(Y) + \sum_{k=1}^{m} w(X_k) & \text{if } Y \text{ is not a leaf} \\ t_{wat}(Y) = t_{rdy}(Y) + w(Y) & \text{if } Y \text{ is neither the root, nor a leaf} \\ t_{wat}(X) = t_{off}(X) + w(X) & \text{if } X \text{ is a leaf} \\ t_{wat}(Y) = t_{rdy}(Y) & \text{if } Y \text{ is the root} \\ \tau_c = t_{rdy}(Y) & \text{if } Y \text{ is the root} \end{cases} \tag{2}$$

We show inductively that S_c and S_d satisfy the two conditions stated in the theorem.

Consider a R with depth $D = 1$. That is, R has a root node A with leaf children $X_1, \ldots, X_m$. For S_d, using Equation 1, we obtain $t_{cpt}(A) = t_d$ and $t_{on}(A) = t_d + \sum_{i=1}^{m} w(X_i)$ which implies

$$t_{on}(A) - t_d = \sum_{i=1}^{m} w(X_i). \tag{3}$$

For child nodes, we obtain $t_{on}(X_i) = t_{cpt}(X_i) = t_d + \sum_{k=1}^{i} w(X_k)$ which implies

$$t_{on}(X_i) - t_d = \sum_{k=1}^{i} w(X_k). \tag{4}$$

Maximization over t_{on}s results in $\tau_d = t_d + \sum_{i=1}^{m} w(X_i)$ and

$$\tau_d - t_d = \sum_{i=1}^{m} w(X_i). \tag{5}$$

From Equation 2 and S_d specified by Equation 3, 4, and 5, we derive S_c as follows:

For each leaf, $t_{off}(X_i) = \tau_d - t_{on}(X_i) + t_c = t_c + \sum_{k=i+1}^{m} w(X_k)$. For the root, $t_{off}(A) = t_{off}(X_m) = t_c$ and $\tau_c = t_{rdy}(A) = t_{off}(X_m) + \sum_{k=1}^{m} w(X_k) = t_c + \sum_{k=1}^{m} w(X_k)$. The above implies

$$\begin{aligned} \tau_c - t_c &= \sum_{i=1}^{m} w(X_i) = \tau_d - t_d \\ \tau_c - t_{off}(A) &= \sum_{i=1}^{m} w(X_k) = t_{on}(A) - t_d \\ \tau_c - t_{off}(X_i) &= \sum_{k=1}^{i} w(X_k) = t_{on}(X_i) - t_d \end{aligned}$$

Therefore, the statement is true when $D = 1$.

Assume that the two conditions in the theorem hold for any R of depth $D = d \geq 1$.

Suppose we add some child nodes to the leaves of R at depth d to form a new tree R'. The depth of R' is $D = d + 1$. We show that the two conditions hold for R' as well. We will add a prime mark (') to a quantity (e.g., t_d, τ_c, and t_{off}) or an object (e.g., S_d and S_c) associated with R' to distinguish it from that associated with R. We will label an equal sign in an equation by the numbers of equations from which the equality is derived, if the derivation may not be obvious to readers.

Let the starting time t'_d of S'_d be the same as t_d of S_d, i.e., $t'_d = t_d$. Then, for each node Y of depth $\leq d - 1$ and each leaf Y of depth d, we have $t'_{on}(Y) = t_{on}(Y)$ which implies

$$t'_{on}(Y) - t'_d = t_{on}(Y) - t_d \tag{6}$$

Let X be a node of depth d in R' with m leaf children: $Z_1, \ldots, Z_m$. Based on Equation 1, we have $t'_{on}(X) = t_{on}(X) + \sum_{j=1}^{m} w(Z_j)$ which implies

$$t'_{on}(X) - t'_d = t_{on}(X) - t_d + \sum_{j=1}^{m} w(Z_j) \tag{7}$$

For each leaf Z_j of depth $d+1$ with parent X, we obtain $t'_{on}(Z_j) = t_{on}(X) + \sum_{i=1}^{j} w(Z_i)$ which implies

$$t'_{on}(Z_j) - t'_d = t_{on}(X) - t_d + \sum_{i=1}^{j} w(Z_i) \tag{8}$$

The termination time $\tau'_d = max(t'_{on}(X)) \geq \tau_d$ where maximization is over all nodes of R'. The above completely specifies S'_d.

We construct S'_c from S'_d based on Equation 2. Let

$$t'_c = t_c - (\tau'_d - \tau_d) \tag{9}$$

which means that S'_c starts earlier than S_c by an amount of time by which S'_d is longer than S_d.

For each leaf Z_j of depth $d + 1$ with parent X, we obtain

$$t'_{off}(Z_j) \stackrel{2}{=} \tau'_d - t'_{on}(Z_j) + t'_c \stackrel{9,8}{=} t_c + \tau_d - t_{on}(X) - \sum_{i=1}^{j} w(Z_i) \tag{10}$$

and

$$t'_{rdy}(X) \stackrel{2}{=} t'_{off}(Z_m) + \sum_{i=1}^{m} w(Zi) \stackrel{10}{=} t_c + \tau_d - t_{on}(X) \tag{11}$$

which implies $t'_{rdy}(X) \stackrel{2}{=} t_{off}(X)$ for each X that is a leaf in R but a parent in R'.

For each leaf at depth $\leq d$,

$$t'_{off}(X) = \tau'_d - t'_{on}(X) + t'_c \stackrel{9}{=} t_c + \tau_d - t'_{on}(X) \stackrel{6}{=} t_c + \tau_d - t_{on}(X) = t_{off}(X) \tag{12}$$

Note that $t'_{rdy}(X)$ of nodes of depth d and $t'_{off}(X)$ of leaves of depth $\leq d$ form a boundary condition for the timing of activities of all nodes above them. Therefore, Equation 11, 12 together with the inductive assumption imply that, for each node of depth $\leq d$, the timing of its activity in S'_c (excluding the inward movement of in-agents from children of a node at depth d) is exactly the same as in S_c. We thus obtain $\tau'_c = \tau_c$ which implies the first condition in the theorem:

$$\tau'_c - t'_c \stackrel{9}{=} \tau_c - t_c + \tau'_d - \tau_d = \tau_d - t_d + \tau'_d - \tau_d = \tau'_d - t_d = \tau'_d - t'_d \tag{13}$$

where the second equality is derived by the inductive assumption.

Finally, we consider $\tau'_c - t'_{off}()$ in terms of four exclusive and exhaustive cases of the node involved:

For each leaf Z_j of depth $d+1$ with parent X,

$$\begin{aligned} \tau'_c - t'_{off}(Z_j) &\stackrel{13,10}{=} (t'_c + \tau'_d - t_d) - (t_c + \tau_d - t_{on}(X) - \sum_{i=1}^{j} w(Z_i)) \\ &\stackrel{9}{=} -t_d + t_{on}(X) + \sum_{i=1}^{j} w(Z_i) \stackrel{8}{=} t'_{on}(Z_j) - t'_d. \end{aligned} \tag{14}$$

For each leaf X of depth $\leq d$,

$$\tau'_c - t'_{off}(X) = \tau'_c - (\tau'_d - t'_{on}(X) + t'_c) \stackrel{13}{=} \tau'_d - t'_d - (\tau'_d - t'_{on}(X)) = t'_{on}(X) - t'_d \tag{15}$$

For each non-leaf node X of depth d,

$$\tau'_c - t'_{off}(X) = \tau_c - t'_{off}(Z_m) \stackrel{10}{=} \tau_c - (t_c + \tau_d - t_{on}(X) - \sum_{i=1}^{m} w(Z_i)) \stackrel{7}{=} t'_{on}(X) - t'_d. \tag{16}$$

For each non-leaf node X of depth $< d$, since $\tau'_c = \tau_c$, $t'_{off}(X) = t_{off}(X)$, $t'_{on}(X) = t_{on}(X)$ and $t'_d = t_d$, we have

$$\tau'_c - t'_{off}(X) = t'_{on}(X) - t'_d. \tag{17}$$

The second condition of the theorem is proven. □

Given a rooted tree R, Theorem 27 implies that, to find the optimal collection schedule in R, we can treat w_{in} for each link as w_{out} for the link, and find the optimal distribution schedule in the modified tree. Once the optimal distribution schedule is found, we can reverse the distribution order and the resultant is the optimal collection schedule. The converse is also correct. We therefore only need to find the optimal schedule for one of the two activities for any given off-line time definition.

Example 28 The upper-left in Figure 10 is a rooted tree R for collection. An optimal collection schedule (with minimal absolute off-line time) can be indirectly obtained by duality.

First w_{in} of each link is treated as w_{out} so the tree becomes one for distribution. Then Algorithm 23 is applied to arrange the left-right order of child nodes for each parent. The resultant is R^1 in the upper right of the figure.

The optimal distribution schedule is specified, according to Theorem 25, as shown in R^2 in the lower left of the figure. The distribution order for each parent is right-to-left.

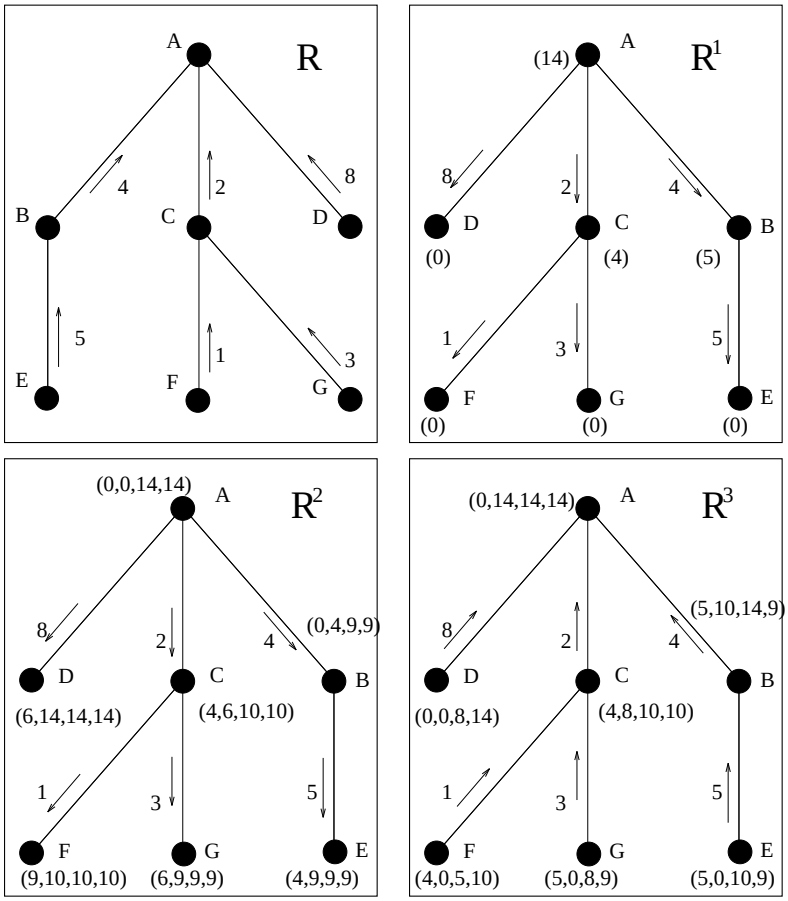

Figure 10: Upper left: A rooted tree R for collection. Upper right: R^1 is converted from R by treating w_{in} as w_{out} and then applying Algorithm 23. Each node X is labeled with $v(X)$ as computed by Algorithm 23. Lower left: R^2 is identical to R^1. The optimal distribution schedule under absolute off-line time is shown by labeling each node X with $(t_{sel}(X), t_{cpt}(X), t_{on}(X), t_{on}(X) - t_d)$ where $t_d = 0$. Lower right: R^3 is converted from R^1 by treating w_{out} as w_{in}. An optimal collection schedule with $t_c = 0$ is obtained from the schedule shown in R^2 as described by Theorem 27. The schedule is shown by labeling each node X with $(t_{off}(X), t_{rdy}(X), t_{wat}(X), \tau_c - t_{off}(X))$.

By treating w_{out} of each link in R^2 as w_{in} again, and reversing the distribution order (from right-to-left to left-to-right), we follow Equation 2 to obtain an optimal collection schedule as shown in R^3 in the lower right of the figure.

Note that we have $\tau_d - t_d = 14 - 0$ in R^2, and $\tau_c - t_c = 14 - 0$ in R^3. Comparing the four-tuples that label corresponding nodes in R^2 and R^3, we see that the last attributes ($t_{on}(X) - t_d$ in R^2 and $\tau_c - t_{off}(X)$ in R^3) have identical values.

In Section 10, we study distribution schedules with the minimal average off-line time, and the result can then be applied to collection schedules by duality.

10 Optimal Schedules with Minimal Average Off-line Time

This section derives the collection and distribution schedules that provide the minimal average off-line time, given a rooted tree.

Let the collection start at time instant $t = t_0$, and terminate at $t = t_1$. Let the distribution start at time instant $t = t_1$, and terminate at $t = t_2$. For each node X, the off-line time is $\Delta(X) = t_{on}(X) - t_{off}(X) = (t_{on}(X) - t_1) + (t_1 - t_{off}(X))$. Note that $t_1 - t_{off}(X)$ is the off-line time of X during collection, and $t_{on}(X) - t_1$ is the off-line time of X during distribution. We therefore obtain

$$\begin{aligned}
\min_{S_c,S_d}(\Delta_{ave}) &= \min_{S_c,S_d}(\frac{1}{N}\sum_{i=1}^{N}\Delta(X_i)) = \frac{1}{N}\min_{S_c,S_d}(\sum_{i=1}^{N}(t_1 - t_{off}(X_i)) + \sum_{i=1}^{N}(t_{on}(X_i) - t_1))) \\
&= \frac{1}{N}(\min_{S_c,S_d}(\sum_{i=1}^{N}(t_1 - t_{off}(X_i))) + \min_{S_c,S_d}(\sum_{i=1}^{N}(t_{on}(X_i) - t_1)))
\end{aligned}$$

where minimization is over all possible collection schedules S_c and distribution schedules S_d. Since minimization over S_d has no effect on the first summation, we have

$$\min_{S_c,S_d}(\Delta_{ave}) = \frac{1}{N}(\min_{S_c}(\sum_{i=1}^{N}(t_1 - t_{off}(X_i))) + \min_{S_c,S_d}(\sum_{i=1}^{N}(t_{on}(X_i) - t_1)))$$

Although minimization over S_c affects the value of t_1, it has no effect on the length of interval $t_{on}(X_i) - t_1$. Thus the second summation is only affected by minimization over S_d. We therefore obtain

$$\min_{S_c,S_d}(\Delta_{ave}) = \frac{1}{N}(\min_{S_c}(\sum_{i=1}^{N}(t_1 - t_{off}(X_i))) + \min_{S_d}(\sum_{i=1}^{N}(t_{on}(X_i) - t_1)))$$

This implies that the optimal collection schedules and the optimal distribution schedules can be studies separately. If we further apply the duality of collection and distribution, we only have to find optimal schedules in general for one of them.

10.1 Optimal distribution schedules

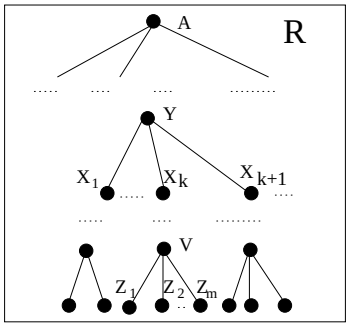

Figure 11: A general rooted tree for distribution

Let R be a tree rooted at A for distribution (Figure 11). Let the distribution operation start at time instant t_1. We denote $\sum(t_{on}(Y) - t_1)$ by $\sigma_R(A)$ where summation is over every node Y of R rooted at A. The optimal distribution schedule is one that minimizes $\sigma_R(A)$.

We first consider the contribution to $\sigma_R(A)$ made by the nodes in a subtree rooted at a parent node, say, V of m leaf children (see Figure 11).

$$\sigma_R(V) = (t_{on}(V) - t_1) + \sum_{k=1}^{m} (t_{on}(Z_k) - t_1).$$

If we substitute $t_{on}(V) = t_{cpt}(V) + \sum_{k=1}^{m} w_{out}(Z_k)$ from Equation 1, we can rewrite

$$\sigma_R(V) \;=\; \sum_{k=1}^{m} w_{out}(Z_k) + (m+1)(t_{cpt}(V) - t_1) + \sum_{k=1}^{m} (t_{on}(Z_k) - t_{cpt}(V)) \qquad (18)$$

The first entry (summation) is independent of the distribution order. The second entry (product) is independent of the distribution order of V. It is dependent of the distribution orders of nodes outside the subtree rooted at V, and is dependent of the number $m+1$ of nodes in the subtree rooted at V. Only the last entry $\sum_{k=1}^{m}(t_{on}(Z_k) - t_{cpt}(V))$, which we denote by $\sigma(V)$, is dependent of the distribution order of V. Note that we have used $\sigma(V)$ instead of $\sigma_R(V)$ to denote the sum, to signify the fact that its minimization can be studied by isolating the subtree rooted at V from the rest of R.

Next, we consider the contribution to $\sigma_R(A)$ made by the nodes in a subtree rooted at an arbitrary non-leaf node Y. (Figure 11).

$$\begin{aligned}\sigma_R(Y) &= (t_{on}(Y) - t_1) + \sum_k \sigma_R(X_k) \\ &= \sum_k w_{out}(X_k) + (t_{cpt}(Y) - t_1) + \sum_k \sigma_R(X_k)\end{aligned}$$

Denote the number of descendants of X_k by n_k, we obtain

$$\begin{aligned}\sigma_R(X_k) &= \sigma(X_k) + (n_k + 1)(t_{cpt}(X_k) - t_1) \\ &= \sigma(X_k) + (n_k + 1)(t_{cpt}(X_k) - t_{cpt}(Y)) + (n_k + 1)(t_{cpt}(Y) - t_1).\end{aligned}$$

If we substitute the above expression into $\sigma_R(Y)$ and denote the number of descendants of Y by η, we obtain

$$\begin{aligned}\sigma_R(Y) &= \sum_k w_{out}(X_k) + (\eta + 1)(t_{cpt}(Y) - t_1) \\ &+ \sum_k \sigma(X_k) + \sum_k (n_k + 1)(t_{cpt}(X_k) - t_{cpt}(Y))\end{aligned} \tag{19}$$

The first entry (summation) is independent of the distribution order. The second entry (product) is dependent of the distribution orders of nodes outside the subtree rooted at Y, and is dependent of the number $\eta + 1$ of nodes in the subtree rooted at Y. The third entry (summation) is dependent of the distribution orders of nodes in the subtree rooted at each child of Y. Only the last entry (summation) is dependent of the distribution order of Y.

Note that equation 18 is a special case of equation 19 if we let $Y = V$, $X_k = Z_k$, $n_k = 0$, $\sigma(X_k) = 0$, and $t_{cpt}(X_k) = t_{on}(Z_k)$.

Another special case of equation 19 is $\sigma_R(A)$. If we let $Y = A$, $t_{cpt}(Y) = t_1$, we obtain

$$\sigma_R(A) = \sum_k w_{out}(X_k) + \sum_k \sigma(X_k) + \sum_k (n_k + 1)(t_{cpt}(X_k) - t_1) \tag{20}$$

The above analysis implies that, in order to find the optimal distribution schedule for R, we need only to search *locally*, i.e., to find the optimal distribution order of each node Y, taking into account the number of descendants of Y.

Lemma 29 *Let Y be a non-leaf node in a rooted tree. Let the child nodes of Y be $X_1, \ldots, X_m$ $(m > 0)$ where X_k has n_k descendants.*

The summation $\psi = \sum_{k=1}^{m} (n_k + 1)(t_{cpt}(X_k) - t_{cpt}(Y))$ is minimized if

$$w_{out}(X_1)/(n_1 + 1) \leq w_{out}(X_2)/(n_2 + 1) \leq \ldots \leq w_{out}(X_m)/(n_m + 1)$$

and distribution order of Y is $O_{out}(Y) = (X_1, \ldots, X_m)$.

Proof:

When $O_{out}(Y)$ is followed, we have $t_{cpt}(X_k) - t_{cpt}(Y) = \sum_{i=1}^{k} w(X_i)$, and therefore

$$\begin{aligned}
\psi &= \sum_{k=1}^{m}(n_k+1)\sum_{i=1}^{k} w(X_i) \\
&= \sum_{i=1}^{1}(n_1+1)w(X_i) + \sum_{i=1}^{2}(n_2+1)w(X_i) + \ldots + \sum_{i=1}^{m}(n_m+1)w(X_i) \\
&= w(X_1)\sum_{i=1}^{m}(n_i+1) + w(X_2)\sum_{i=2}^{m}(n_i+1) + \ldots + w(X_m)\sum_{i=m}^{m}(n_i+1)
\end{aligned}$$

where we have written $w(X_i)$ instead of $w_{out}(X_i)$ for simplicity.

Assume that the following condition holds:

$$w(X_1)/(n_1+1) \leq w(X_2)/(n_2+1) \leq \ldots \leq w(X_m)/(n_m+1)$$

When $m = 2$, $\psi = w(X_1)(n_1+1+n_2+1) + w(X_2)(n_2+1)$ if $O_{out}(Y)$ is followed. If instead the other possible order $O'_{out}(Y) = (X_2, X_1)$ is followed, using the similar derivation, we have $\psi' = w(X_2)(n_1+1+n_2+1) + w(X_1)(n_1+1)$. The difference $\psi' - \psi = w(X_2)(n_1+1) - w(X_1)(n_2+1) \geq 0$ since $w(X_1)/(n_1+1) \leq w(X_2)/(n_2+1)$ by assumption.

In general, for each X_k, ψ contains exactly m X_k-related addends, we denote their sum by P:

$$\begin{aligned}
P &= (n_k+1)w(X_1) + \ldots + (n_k+1)w(X_{k-1}) + (n_k+1)w(X_k) \\
&+ (n_{k+1}+1)w(X_k) + \ldots + (n_m+1)w(X_k)
\end{aligned}$$

Note that each addend in P (in the form $(n_i+1)w(X_j)$) is unique.

Given an arbitrary distribution order $O'_{out}(Y) = (X_{1'}, \ldots, X_{m'})$ where $i' \in \{1, \ldots, m\}$ and where $l' = k$, i.e., X_k appears at lth location in $O'_{out}(Y)$, we also have exactly m X_k-related addends in ψ'. we denote their sum by P':

$$\begin{aligned}
P' &= (n_k+1)w(X_{1'}) + \ldots + (n_k+1)w(X_{l-1'}) + (n_k+1)w(X_k) \\
&+ (n_{l+1'}+1)w(X_k) + \ldots + (n_{m'}+1)w(X_k)
\end{aligned}$$

Note that each addend in P' is also unique.

We show the $P' - P \geq 0$ for every X_k. To show that, it is sufficient to create an one-to-one correspondence f from the set of addends of P' to the set of addends of P such that $p' - p \geq 0$ where p' is an addend of P' and $p = f(p')$.

A typical addend of P' is either in the form $(n_k+1)w(X_{i'})$ or in the form $(n_{i'}+1)w(X_k)$.

Suppose $p' = (n_k+1)w(X_{i'})$. If $i' < k$, there exists a unique $p = (n_k+1)w(X_{i'})$ in P and $p' - p = 0$. If $i' \geq k$, there exists a unique $p = (n_{i'}+1)w(X_k)$ in P and $p' - p \geq 0$ since $w(X_{i'})/(n_{i'}+1) \geq w(X_k)/(n_k+1)$ by assumption.

Suppose $p' = (n_{i'} + 1)w(X_k)$. If $i' < k$, there exists a unique $p = (n_k + 1)w(X_{i'})$ in P and $p' - p \geq 0$ since $w(X_{i'})/(n_{i'} + 1) \leq w(X_k)/(n_k + 1)$ by assumption. If $i' \geq k$, there exists a unique $p = (n_{i'} + 1)w(X_k)$ in P and $p' - p = 0$.

Therefore, $P' - P \geq 0$ for every X_k, and the order $O_{out}(Y)$ minimize ψ. □

Theorem 30 (Optimal distribution schedule)

Let R be a weighted tree rooted at A. For each non-leaf node Y of R, let the $m > 0$ child nodes of Y be indexed from left to right as $X_1, \ldots, X_m$, and let the number of descendants of X_i be n_i such that

$$w_{out}(X_1)/(n_1 + 1) \leq w_{out}(X_2)/(n_2 + 1) \leq \ldots \leq w_{out}(X_m)/(n_m + 1).$$

The average distribution off-line time $\sigma_R(A)$ is minimized if the distribution order of Y is left-to-right for every Y.

Proof:

First, consider the case where the depth of R is $D = 1$. Let the child nodes of root A be $X_1, X_2, \ldots$.

According to Equation 20, when $D = 1$

$$\sigma_R(A) = \sum_k w_{out}(X_k) + \sum_k (t_{cpt}(X_k) - t_1).$$

To minimize $\sigma_R(A)$, we only need to minimize the second sum. According to Lemma 29, $\psi = \sum_k (t_{cpt}(Y_k) - t_1)$ can be minimized if the left-to-right order is followed. The statement is thus true when $D = 1$.

Assume that the statement is true when the depth of R is $D = d \geq 1$. That is, $\sigma_R(A)$ is minimized when the left-right order of child nodes is arranged as specified and the left-to-right distribution order is followed.

We consider

$$\sigma_R(A) = \sum_k w_{out}(X_k) + \sum_k \sigma(X_k) + \sum_k (n_k + 1)(t_{cpt}(X_k) - t_1)$$

when $D = d + 1$. Based on the analysis made with Equation 19, minimization of each $\sigma(X_k)$ and minimization of $\psi = \sum_k (n_k + 1)(t_{cpt}(X_k) - t_1)$ are independent, and can thus be performed separately. According to Lemma 29, ψ is minimized if the left-to-right order is followed, Each $\sigma(X_k)$ can be minimized by following the left-to-right order according to the inductive assumption. The statement is proven. □

Example 31 Figure 12 shows an optimal distribution schedule obtained according to Theorem 30. We can compare this schedule with the distribution schedule in Example 22. There, no preferred distribution order for node C and $O_{out}(C) = (F, G)$ is arbitrarily chosen. The consequence is $\sigma_{R'}(A) = 16 + 9 + 14 + 16 + 7 + 9 + 14 = 85$. Here,

the preferred distribution order for node C is $O_{out}(C) = (G, F)$. The consequence is $\sigma_{R'}(A) = 16 + 9 + 14 + 16 + 5 + 9 + 14 = 83$.

This comparison shows that an optimal schedule under the absolute off-line time criterion may not be optimal under the average off-line time criterion. However, for the absolute off-line time, both examples happen to have the identical value 16.

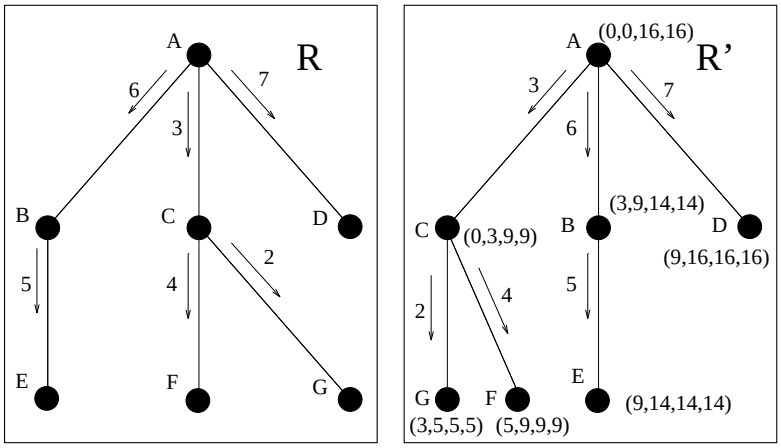

Figure 12: R: A rooted tree for distribution. R': R with the left-right order of nodes B, C, D rearranged according to Theorem 30. The distribution schedule is shown by the label $(t_{sel}(X), t_{cpt}(X), t_{on}(X), t_{on}(X) - t_d)$ at each node X. The distribution starts at $t_d = 0$.

Example 32 Figure 13 shows two distribution schedules in an identical rooted tree (subject to a difference in left-right order of child nodes). The schedule shown in R is optimal under the absolute off-line time criterion, obtained according to Theorem 25. The distribution order is right-to-left. It has $\Delta_{1-2} = 14-0$ and $\sigma_R(A) = 9+14+12+14+12+8 = 69$. The schedule shown in R' is optimal under the average off-line time criterion, obtained according to Theorem 30. The distribution order is left-to-right. It has $\Delta_{1-2} = 15 - 0$ and $\sigma_R(A) = 9 + 8 + 15 + 8 + 11 + 15 = 56$.

Example 32 shows that, given a rooted tree, an optimal schedule under the absolute off-line time criterion may not be optimal under the average off-line time criterion, and vice versa. Though the example involves distribution only, the conclusion is general due to the duality of distribution and collection.

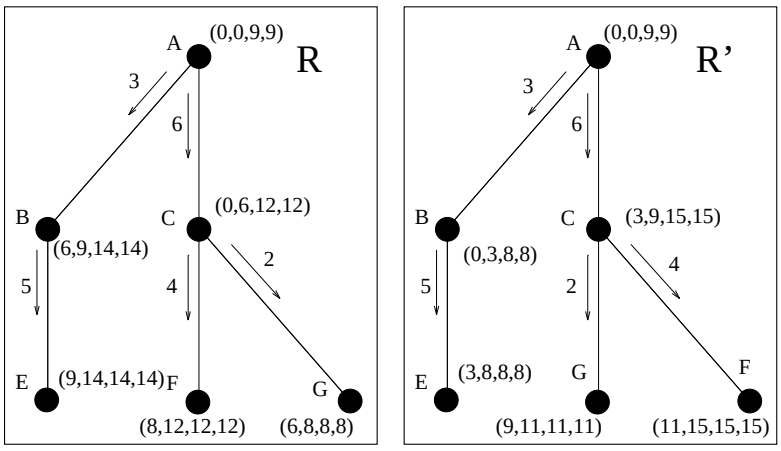

Figure 13: Comparison of optimal distribution schedules under different criteria. An schedule with the minimal absolute off-line time is shown in R. An schedule with the minimal average off-line time is shown in R'. Each schedule is shown by the label $(t_{sel}(X), t_{cpt}(X), t_{on}(X), t_{on}(X) - t_d)$ at each node X in the corresponding rooted tree. The distribution starts at $t_d = 0$.

11 Discussion

11.1 DAI, tree structure and conditional independence

In a MSBN, the interface between any pair of subnets satisfies a d-sepset condition such that, in order to make neighbours up-to-date, it is sufficient to pass the up-to-date probability distribution on the interfacing set between them and nothing else.

Each subnet in a MSBN is transformed into a single JT. The tree structure maintains a single information path between each pair of nodes in the JT such that local belief updating can be performed efficiently. The price paid is the multiple information paths between neighbour JTs: Passing evidence from one JT to a neighbour requires computation proportional to the number of linkages between them.

Such built-in preference of the efficiency of local computation over the efficiency of communication is consistent with the general assumption in multi-agent intelligent systems. In DAI, we assume that the distance between agents prevents constant communication. Therefore, each agent must rely on its local resource in its problem solving, and can communicate with other agent only occasionally.

One of the major concerns in DAI is how to balance the need to maintain as much as possible global consistency and the need to reduce the traffic of communication. As argued by Pearl [10], a tree structure makes use of causal independence and allows the most

efficient information passage among a group of elements. Our study on MSBNs highlights the MSBNs that are organized into a *hypertree* structure. This structural preference is a more general case of the exploration of the causal independence in singly connected BNs [9] and in the JT representation of multiply connected BNs [4]. If we call an element which can render a pair of elements conditionally independent a 'causal mediator', we see an increase of complexity of the internal structures of causal mediators in the three cases. In singly connected BNs, a causal mediator is an internal node in the network. In the JT representation of multiply connected BNs, a causal mediator is an internal clique (a group of variables). In MSBNs/LJFs of hypertree structure, a causal mediator is a subnet/JT. As argued by Pearl [10], conditional independence should not be viewed as a restrictive assumption for mathematical convenience, nor as an occasional grace of nature for which we must passively wait, but rather as a mental construct that we should actively create. The progression of probabilistic reasoning techniques from singly connected BNs, to the JT representation of multiply connected BNs, and to MSBNs/LJFs is just one example of such endeavor.

11.2 Communication in MSBNs and belief propagation in a JT

Our **CommunicateBelief** operation in a multi-agent MSBN is a direct extension of **CollectionEvidence** and **DistributeEvidence** in a single JT by Jensen et al.[4, 5], and multi-linkage belief propagation in a single-agent MSBN by Xiang et al.[12].

If we concentrate on the overall pattern of information flow, we see that communication in a MSBN works in the same way (a semi-parallel inward movement followed by a semi-parallel outward movement in a tree structure) as belief propagation in a JT. As commonly applied, BNs are single-agent-oriented. Therefore, all nodes in a JT being centralized is taken granted. The special information flow pattern is used to support object-oriented implementation as indicated in Jensen et al.[5], but concurrent processing is reasonably considered uninteresting.

However, in a multi-agent MSBN, the agents are very likely to be spatially distributed, the efficiency in communication becomes a necessary concern. Half of this paper has thus been devoted to this issue.

Once the issue in concurrency control is resolved, however, the result can be applied to the belief propagation in a JT, if parallel processing of a JT is desired and the hardware is available.

11.3 Hybrid systems

In this pilot study, we propose a probabilistic framework for multi-agent distributed interpretation systems, and we show how consistency among agents can be maintained and how communication can be optimized to reduce the unavailability of agents for evidence

processing. We make no claim that every agent in the system must be a BN. An agent may take other representation formalism, and integration of a hybrid system is one of the issues that demand further study. A study of integrating inexact reasoning models (other than BNs) in DAI systems is presented by Zhang [13].

Acknowledgement

This work is supported by the Dean's Research Funding from Faculty of Science, University of Regina, the General NSERC Grant from University of Regina, and Research Grant OGP0155425 from NSERC.

References

[1] A.H. Bond and L. Gasser. An analysis of problems and research in dai. In A.H. Bond and L. Gasser, editors, *Readings in Distributed Artificial Intelligence*, pages 3–35. Morgan Kaufmann, 1988.

[2] A.H. Bond and L. Gasser, editors. *Readings in Distributed Artificial Intelligence.* Morgan Kaufmann, 1988.

[3] L. Gasser and M.N. Huhns, editors. *Distributed Artificial Intelligence, Volume II.* Morgan Kaufmann, 1989.

[4] F.V. Jensen, S.L. Lauritzen, and K.G. Olesen. Bayesian updating in causal probabilistic networks by local computations. *Computational Statistics Quarterly*, (4):269–282, 1990.

[5] F.V. Jensen, K.G. Olesen, and S.K. Andersen. An algebra of bayesian belief universes for knowledge-based systems. *Networks*, 20:637–659, 1990.

[6] S.L. Lauritzen and D.J. Spiegelhalter. Local computation with probabilities on graphical structures and their application to expert systems. *Journal of the Royal Statistical Society, Series B*, (50):157–244, 1988.

[7] V.R. Lesser and L.D. Erman. Distributed interpretation: a model and experiment. *IEEE Transactions on Computers*, C-29(12):1144–1163, 1980.

[8] R.E. Neapolitan. *Probabilistic Reasoning in Expert Systems.* John Wiley and Sons, 1990.

[9] J. Pearl. Fusion, propagation, and structuring in belief networks. *Artificial Intelligence*, (29):241–288, 1986.

[10] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 1988.

[11] Y. Xiang, B. Pant, A. Eisen, M. P. Beddoes, and D. Poole. Multiply sectioned bayesian networks for neuromuscular diagnosis. *Artificial Intelligence in Medicine*, 5:293–314, 1993.

[12] Y. Xiang, D. Poole, and M. P. Beddoes. Multiply sectioned bayesian networks and junction forests for large knowledge based systems. *Computational Intelligence*, 9(2):171–220, 1993.

[13] C. Zhang. Cooperation under uncertainty in distributed expert systems. *Artificial Intelligence*, (56):21–69, 1992.