Chapter 3

Planning Alternatives for ONA

Planning problems are often fairly complicated. Planning for ONA adds more complexity to this problem because of its real-time nature. In the next section we will show that the space of solutions grows exponentially with the number of nodes in a connection. The hardness of the problem is NP-complete, and thus will require a heuristic solution. We will show various aspects of the planning problem and describe some approaches to its solution in subsequent sections.

3.1 The Requirements of Planning

Planning consists of an action selection phase where actions are selected and ordered to reach the desired goals, and a resource allocation phase where enough resources are assigned to ensure the successful execution of the chosen actions. An ONA plan is a set of instructions to the nodes participating in a connection concerning what adapters to use and in which order and where. The calculation of a plan depends on many factors and is computationally complex. We will consider the most critical problems below.

3.1.1 Temporal factor

Real-time applications require a very fast connection establishment, so that a plan can be created relatively quickly, since the data stream cannot be delayed indefinitely in search of the perfect plan. Depending on the specifics of the data stream, between microseconds and very small numbers of seconds are available to plan the remedial strategy. If the code implementing remedial action is not ubiquitous, deployment costs must also be considered in planning.

The shorter the connection life, the faster the planning process must be. A plan space of possible solutions for a particular connection can be very big, but the duration of the search within that space should fit the boundaries of communication-establishment time limits. The search should rely on some heuristics that optimize for the most common cases. Using these heuristics simplifies the search, but at the same time it can cause the loss of good plans, which leads to poorer plans than one would have with the exhaustive search. On-line heuristic planning presumes a possibility that a solution might not be found, in which case the connection cannot satisfy the quality of service requirements or simply fails. The probability of a failure must be below the threshold of acceptable risk.

This temporal constraint plays the role of a global limit to the plan calculation process. If no feasible plan was calculated within temporal limits it means that the planning process failed. Otherwise the best feasible plan found must be returned even if much better plans might have been evaluated. In the case of a very strict temporal constraint, the whole planning process can be stopped and an incompletely calculated

plan returned as the only possible global plan. For a sufficiently long communication session, a cheap and inefficient preliminary plan can be calculated and deployed, and the data transfer started. The search for a better global plan can continue in the background, and an optimized global plan can be deployed later.

3.1.2 The consistency of adaptations

The system must sufficiently understand the format of the data stream that it intends to improve in order to take proper actions. In some cases, not only the format of the data stream must be considered, but also application end-points, hardware devices at those endpoints, and even wishes and needs of users. Certain adapters are format-aware, such as distillers, adapters that modify the actual content of user data. For example, a colored image in format IMG must be transferred from node A to node B through an extremely poor link. Unfortunately, our distiller can understand only JPEG format. The planner must recognize that it should apply an IMG-JPEG converter first, then apply the distiller, and convert the data to the original format using a JPEG-IMG converter. The planner must analyze data format consistency and apply format conversion whenever necessary.

3.1.3 The ordering of adapters

The stream may encounter multiple problems at various points along the transmission path, and generally different actions will be required to solve those problems. Applying multiple actions implies that the system must be able to determine whether a set of actions is compatible and effective. The canonical example of

incompatibility is to meet problems of security and inadequate bandwidth by first encrypting the data stream, then ineffectually compressing the encrypted version of the stream. The difference between this case and the case of data format consistency is that in the latter case, there is no conflict of data formats. Compression can be correctly applied to encrypted data; it simply fails to achieve its goals. The only problem is the ineffectiveness of using the adapters in one order and the effectiveness of using them in the opposite order. This problem requires an extension of the notion "format" that expresses the "compressibility" of data. As compressibility of data decreases significantly after encryption, the planner should make the decision to put the compressing adapter before encryption. The compressing adapter also reduces the compressibility of data, but the encryptor is indifferent to this aspect of data format (unlike the compressing adapter) and will be effective. Understanding these aspects of data format is very important for planning. The number of and interactions with such aspects may change over time, so the planner should be as independent of them as possible.

3.1.4 User preference as an element of planning

Some user applications have extra requirements for data streams. For example, palmtops have the ability to receive and process images in only n certain formats. These applications may require special treatment of data stream on intermediate nodes, i.e., conversion of the data from the original format to the one that is appropriate for the palmtop. ONA allows the automatic deployment of special format-converting protocols serving the particular needs of that palmtop. Another example is filters that can be

applied to video data streams. If a stream need to be filtered, a user can choose to drop color but preserve the resolution of the video, or conversely, to give up a higher resolution but keep color.

User preferences can be obtained by an ONA planner through some API or configuration mechanism.

3.1.5 The efficiency of the plan

It is important to save the execution power of network nodes and links. Adapters should be applied in the most efficient way, i.e., without repetitions, minimizing the use of node and link resources. For example, consider a connection that consists of three nodes A, B, and C, connected with two links AB and BC. Assume that link AB requires two-fold compression of data, and link BC requires four-fold compression of data (see Fig. 3.1). The optimal plan should apply compression only once, at node A with four-fold compression; this will satisfy the requirements for all links without requiring a wasteful compression and encryption.

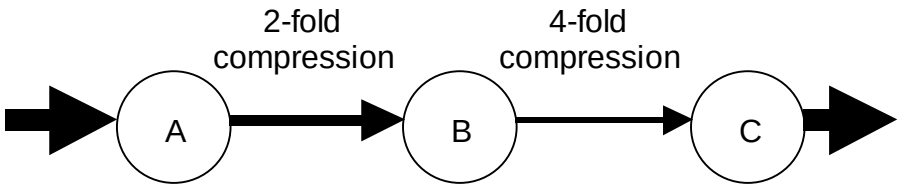

Fig. 3.1: Two-fold and four-fold data compression

Another way to increase the efficiency of adapters would be to extend the effects of an adapter that reduces the amount of resources needed for the connection. For example, a peer-to-peer connection consists of a number of nodes and links that are connected. One of the intermediate links requires compression of the data. The whole network can benefit if the compression will be run on the source and decompression on the destination of this connection, because of the totally reduced amount of data transferred. The problem is that the system must be able to determine if the open architecture is willing and able to run all adapters that the system proposes at the locations it chooses. Some adapters might not run properly on particular nodes. Some nodes might be unwilling to run particular kinds of adapters. Perhaps some nodes limit the resources expendable on a single packet or entire data flow at that node. These constraints may cause a different set of adapters to be chosen, or may affect where adapters are located.

Optimization of a plan presumes an extensive search in the space of possible plans. The plan that demonstrates the best use of network communication and execution resources must also be deployable. A plan that is deployable is called *feasible*. As we have shown in the previous example, the best location of a compression adapter would be the end points of the connection. But the plan might not be feasible if the endpoints of the connection are unable to deploy the adapter because of insufficient resources, lack of access to the chosen adapters, etc. The costs of the deployment of a plan can be another factor of optimization.

The planner also should take into account the fact that each adapter applied to a data stream adds delay to that stream. Even if every adapter alone does not trespass the threshold of latency of data transfer, all together they can make the latency unacceptably long. In such cases the planner should try further compression of the transferred data, choose less time-consumptive adapters, run adapters on more powerful servers, etc.

3.1.6 The extensibility of the system

Extensibility of planning is a serious problem for the planning system. New transformations, data formats, and constraints that are unknown today can appear in the future. The need to handle both existing network problems and future network problems will produce a significant number of different adapters, written by many parties. The design of the planner and adapters should presume some common interface, well understood by both groups of designers. Creating or using a plan requires knowledge of the available adapters. The planner should know their names, locations, and how to use them. The planner should distinguish between the versions and the specifics of adapters. The planner should know the amount of resources that those adapters require.

If ONA planning cannot accommodate these new components and combine them with older components properly, the automated planning approach is of much less value. The only way to use an auto-planning approach is to augment each planner with its own libraries of adapters. It would make the life of a planner designer much easier, but we believe that in practice, the adapters and planner will be maintained independently.

3.1.7 Resource management for planning

Resource management is serious problem for ONA planning. Some projects offer resource reservation systems for ONA-enabled networks [Bush99] and [Braden01]. The problem is that the planner cannot tell in advance what resources will be necessary for a connection. But during the period of plan calculation, information about the networks can change because other sessions involving some of connection nodes can be started. Then parts of the resources that the newly calculated plan was determined to use are not available any more. These resources cannot be reserved for the session and the plan becomes obsolete. Therefore, a more advanced resource reservation technique is necessary to handle this problem.

The source node can participate in a number of connections for which it is either the source or the destination node. If a new connection must be established, the source node can have the power to terminate old connections or fairly redistribute network resources among old connections and the new connection. Thus, the planning system should count the source node resources used by the connections and replan them for resource redistribution whenever necessary.

3.1.8 Other problems

Errors can occur during the process of planning, deployment or running the adapters. The planner should be able do error handling to preserve the safety of data and rerun the planning process.

The composition of adapters also raises security problems. Some adapters will require that the user be authorized to execute the adapter. In some cases the process of

planning will require that the designer of the adapter be trusted. Resources necessary for deployment of the plan may be subject to secure access constraints.

Another problem is that the use of some adapters will need to be monitored for accounting purposes. We believe that all these issues can be handled by planner design in the future.

3.2 Complexity of Planning Problem

3.2.1 Physical model

ONA-enabled nodes deploy adapters that modify a user's data stream to allow it to achieve better communications between a source and destination. The links that connect the nodes can be described by bandwidth, jitter, security, monetary cost of use, etc. User data streams are described by properties such as format, required throughput, encryption, etc. ONA nodes have limited computational resources to run adapters, and user preferences introduce other constraints. The goal of planning is to distribute adapters on nodes of a connection, respecting all constraints, and often compensating for link inadequacies by using extra node resources.

Adapters typically consist of one or two parts. For example, a filter or a format converter is an unary adapter; encryption and decryption are two parts of a binary adapter. Adapters can be sensitive to data format, the order of adapters, the location, etc. Some adapters convert the form of data but preserve it; some adapters achieve their goal through dropping or transforming part of the data. Different adapters may require different computation resources.

In addition to respecting other constraints, the planner must understand the data stream format to improve it. Some adapters are format-aware, e.g., distillers that modify the actual content of user data.

Because the data stream may encounter multiple problems at various points along the transmission path, the planner must be prepared to deploy multiple remedial actions, which implies that the system must be able to determine whether the actions are compatible. For example, if one adapter converts a color video stream to black and white, there is no value to applying another adapter that drops the color depth of the original stream.

We wish to minimize the resource costs of running adapters, which implies that we want to avoid unnecessary duplication of adapters, running adapters that don't actually help, and other wasteful actions.

The planning system must be extensible to allow it to handle new data transformations, data formats, and constraints. The planner and adapters should share a common interface to ease extensibility. The planner should be aware of adapters' names, locations, costs, and how to use them.

Real-time applications require a very fast connection establishment, so the plan must be created relatively quickly. The data stream cannot be delayed indefinitely in search of the perfect plan.

3.2.2 Observation of the planning problem

Assume that we have a unicast connection between source, destination, and a number of intermediate nodes (see Figure 3.2):

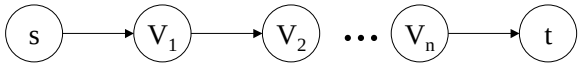

Figure 3.2: A unicast connection

.

Connection resources can be expressed as the following:

connection resources = link resources + node resources

The planner seeks to trade off the node and link resources using ONA adapters to meet the requirements of the user applications:

1. The link resources used must be below the network limits.
2. The node resources used must be below the node limits; they should be minimal but sufficient to guarantee 1.
3. Response times must be below the limit.

Link resources include bandwidth, reliability, security, monetary cost, etc. Node resources include CPU cycles, memory, hard drive space, cost, etc. Each connection must use less of a resource than the amount provided by the network.

3.2.3 Estimation of the space of solutions

Let us try to make a rough estimation of the space of possible plans that refer to all possible adapter positions on the nodes of a connection.

Assume that we have a connection with N nodes. L is the number of nodes that are designated to run adapters. P is a number of chosen adapters that handle problems of the connection. Assume that each adapter can consist of two parts: DO and UNDO. For example, the encryption part of an adapter corresponds to the DO part, while the decryption part corresponds to the UNDO part.

Consider the case where all network problems occur on the end link of the connection, at either the source or destination node. Let us assume that all our problems occur on the first link of the connection as shown on Figure 3.3a; then all DO parts must be run on the very first node, while the left to right UNDO parts can be distributed among L nodes. We assume that all adapters are already ordered in the sequence in which they must be executed. Then the number of possible L nodes that run adapters is equal to $\binom{N}{L}$, the number of possible distributions of the adapters among L nodes is equal to $\binom{P+L}{P}$. Therefore, the space of the solutions for this case is:

$$P(N, L, P) = \binom{N}{L}\binom{P+L}{P},\ P \geq L,\ L \leq N.$$

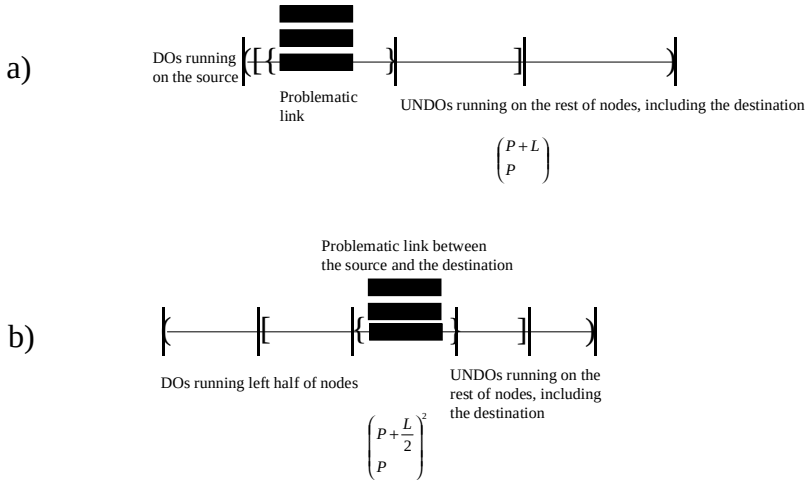

Figure 3.3: The estimation of the solution space: a) problems are located at the link that is adjacent to the source; b) problems are located on the link that is in the middle of the path between the source and destination.

If we locate the link with all problems at the middle link of the connection as Figure 3.3b presents, the space of solutions will be equal to:

$$P_1(N, L, P) = \binom{N}{\frac{L}{2}}^2 \binom{P+\frac{L}{2}}{P}^2, \; P \geq L, \; L \leq N.$$

The DO and UNDO parts of adapters follow the same rules as the parenthesis of algebraic expressions: if the DO part of one adapter is on the left side of the DO part of another adapter, then the UNDO part of the former adapter must be on the right side of the latter adapter. This rule reduces the number of possible locations of the adapters and can be approximated with *Catalan* numbers [Roberts84] that estimate all possible positions for the parenthesis in algebraic expressions. Then values P_1 and P can be expressed:

$$P(N, L, P) = \binom{N}{L}\binom{P+L}{P} \geq \left(\frac{N}{L}\right)^{L}\left(\frac{P+L}{P}\right)^{P},$$

$$P_1(N, L, P) = \binom{N}{\frac{L}{2}}^{2}\binom{P+\frac{L}{2}}{P}^{2} \geq \left(\frac{2N}{L}\right)^{2L}\left(\frac{2P+L}{2P}\right)^{2P}.$$

It is easy to see that expressions for P_1 and P are both exponential and $P_1 >> P$. We can deduce that if problems occur on the links other than end links, it increases the space of the possible solutions. Hence, value P, which is exponential, can be used as a minimal solution space for our estimation. The fact that the solution space is bigger than some exponential dependency implies that it is exponential itself.

3.2.4 NP-completeness

Our goal is to show that the task of finding an optimal sequence of adapters is NP-complete. We will show that a simplified partial case of our problem is similar to a well-known NP-complete problem [Cormen86]. Specifically, we show that our task can be transformed from the shortest weight constrained path problem [Garey79].

The proof is the following.

Let us show that the solution of our problem will also solve the shortest weight constrained path problem.

Let us reformulate our problem assuming that any plan that we can build for our connection defines a new connection with a particular link capacity and cost. The source *s* and destination *t* are connected with a number of alternative paths (see Figure 3.4).

Plans can share nodes and links in cases when a node runs exactly the same adapters and forwards the identical data flows for different plans.

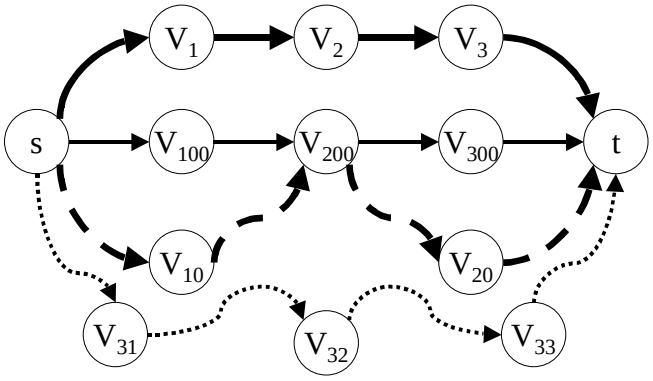

Figure 3.4: Peer-to-peer connection as a group of alternative plans

Different nodes on the diagram can represent the same physical nodes that run different sets of adapters. For example, assume that we have a connection that consists of 3 nodes A, B and C, as shown in Figure 3.5a. Assume that link AB requires compression and encryption of the user data, and link BC requires just compression. Figure 3.5b shows alternative plans that can be built for this connection. Figure 3.5c shows the directed graph that represents these plans. Nodes *s* and *t* are added to the graph assuming that all resources of s, t, and the resources of the "links" that connect them to the graph are equal to 0.

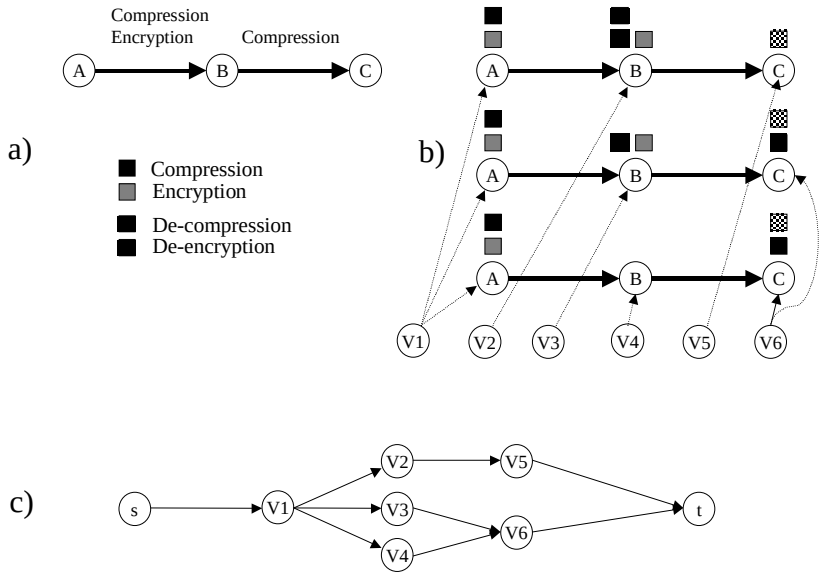

Figure 3.5: Alternative plans for a three-node connection: a) connection b) three alternative plans for the connection c) graph representation of alternative plans

The number of possible plans that defines the number of alternative paths depends exponentially on the number of links and the number of network problems that influence the connection. In the previous section we claimed that the number of possible plans for a connection that consists of N links, suffers P problems, and runs P adapters on L nodes is at least $\binom{N}{L}\binom{P+L}{P}$. This number represents the scale of the space of plans for a connection that is exponential.

Let us combine link and node costs such that the whole variety of link and node characteristics is reduced to only two: packet delay and delivery cost. For example, assume we have a composition with three nodes (A, B, and C) and two links (AB, BC). The first link is a low-bandwidth and requires compression/decompression, the second is insecure and requires encryption/decryption.

Then delay can be created by:

- Compression latency *(cl)*
- Packet delivery per link (*pl*),
- Compressed-packet delivery per link (*cpl*)
- Packet delivery through low bandwidth link (*PL*)
- Compressed-packet delivery per low bandwidth link (*CPL*)
- Decompression latency (*dl*)
- Encryption (e)
- decryption (*d*)

The cost can be determined by:

- Cost associated with the cost of running adapter (*a*)
- Cost associated with the cost of running an adapter on a node that does not have enough resources to run the adapter ( )
- Insecurity of not encrypted data on insecure link (*I*)
- Insecurity of unencrypted data on an secure link (*i*)
- Insecurity of encrypted data on an insecure link (*IE*)
- Insecurity of encrypted data on a secure link (*ie*)

Assuming inequities,

$PL >> pl >> cpl$

$PL >> CPL >> cpl$

$I >> IE >> ie$

$I >> i >> ie$

All possible paths among plan routes have the following latencies and costs:

a) *Latency = PL + pl, Cost = i + I;*

b) *Latency = cl + CPL + dl + pl, Cost = i + I +2a;*

c) *Latency = cl + CPL + cpl + dl, Cost= i + I +2a;*

d) *Latency = PL+ e + pl + d, Cost = i + IE + 2a;*

e) *Latency = cl + CPL + dl + e + pl + d, Cost = i + IE + 4a;*

f) *Latency = cl + CPL + e + cpl + d + dl, Cost = i + IE + 4a;*

g) *Latency = cl + e + CPL + cpl + d + dl, Cost = ie + IE + 4a;*

h) *Latency = cl + e + CPL + d + dl + e + pl + d, Cost = ie + IE + 6a.*

Thus, plan *a)* contains no adaptation. Plan *b)* compresses the data on A and decompresses on B. Plan *c)* compresses data on A and decompresses on C. Plan *d)* encrypts on B and decrypts on C. Plan *e)* compresses on A and decompresses on B, encrypts on B and decrypts on C. Plan *f)* compresses on A and decompresses on C, encrypts on B and decrypts on C. Plan *g)* compresses and encrypts on A and decompresses and decrypts on C. Plan *h)* compresses and encrypts on A, decompresses on B and decrypts on C.

Then the problem can be formulated as the following:

Directed graph combines plan paths from *a)* to *h)* as it was described earlier. Each plan path is associated with the latency and the cost.

Find the plan represented by the path with the constraints:

$Latency \leq cl + CPL + e + cpl + d + dl;$

$Cost \leq ie + IE + 4a.$

The condition on *Latency* is satisfied by *f)* and *g).* The condition on *Cost* is satisfied by only *g).* Thus, the solution that satisfies the conditions above is *g).*

Note that the values *PL* and *I* are assumed high to penalize the planning solution for the connections where the constraints on bandwidth and security are not satisfied. In the simple cases when the problem solution is binary, *adapter applied* or *adapter not applied*, these values can be assumed *infinity*.

Thus, assuming:

1. All link and node characteristics are transformed to just two characteristics: packet delay and delivery cost (as was done above).
2. Packet delay is a sum of per link delays that characterize the quality of predicted service [Clark92], limited by bound W.
3. The cost of the connection is limited by bound K.

the problem is reformulated as the following:

INSTANCE: Directed graph $G=(V, E)$, time $l(e) \in Z^+$, and cost $w(e) \in Z^+$ (where Z^+ denotes the positive integers), for each $e \in E$, specified vertices $s, t \in V$, positive integers K, W.

QUESTION: Is there a simple path G from s to t with total time W or less and total cost K or less?

This problem is similar to the shortest weight constrained path problem [Garey79]. Thus, the solution to this problem will also resolve the shortest weight constrained path problem for the case of the directed graph, which is transformed from the partition problem. The partition problem is NP-complete, so the shortest constrained path problem is NP-complete too.

Therefore, building a plan for a network connection is an NP-complete problem.

3.2.5 Insufficiency of exhaustive search

As shown in the previous sections, the solution space is exponential, and the hardness of the planning problem is NP-complete. This implies that exhaustive search is not applicable to connections that contain a large number of nodes and use a large number of adapters. The complexity of the exhaustive search is the same as the complexity of the solution space, thus exponential. The exhaustive search in this space is not possible for real-time applications that will have to wait an undetermined number of seconds, minutes, or even hours for the solution to the planning problem. Another solution to the planning problem must be found.

3.3 Naïve Planning

We showed in the previous section that an exhaustive search solution is not suitable for our kind of connections. However, naïve approaches that provide primitive heuristic solutions may not be suitable for the needs of our connections, either. In examples of such solutions, all necessary adapters are put either at the nodes that

surround problematic links or at the end nodes of the connection. Examples of this approach are presented in Figure 3.6.

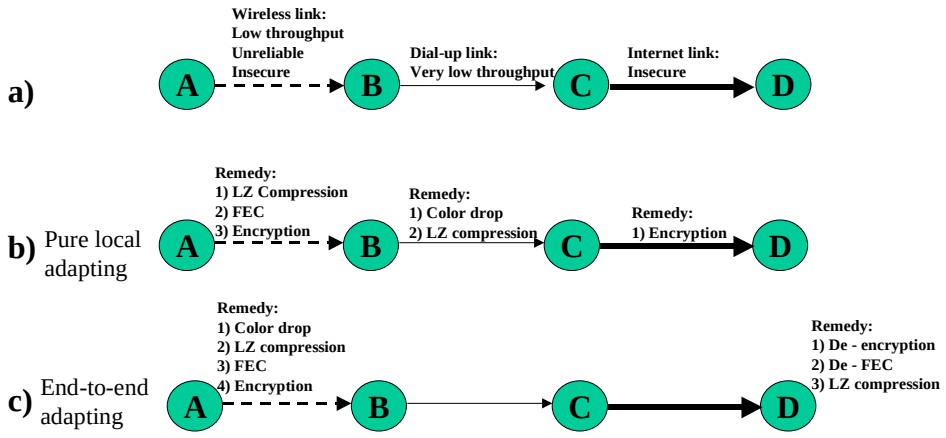

Figure 3.6: Examples of naïve solutions

Figure 3.6a presents the connection of four nodes: A, B, C, and D. Assume link AB is wireless and thus low-bandwidth, insecure and unreliable. Assume link BC is a dial-up link and thus has very low bandwidth. Assume link CD is an Internet link and thus insecure.

Figure 3.6b presents the solution for this connection by applying adapters to the links where the corresponding problems occurred. Lempel Ziv compression, encryption, and forward error correction (FEC) adapters are applied to link AB. A filter and Lempel Ziv compression are applied to link BC. Encryption is applied to link CD. Thus, we run Lempel Ziv compression and encryption twice. This choice causes unnecessary latency

of data and overuses the computational resources of nodes that could serve other connections instead.

Figure 3.6c shows the insufficiency of the opposite solution, applying the adapters to end nodes of the connection. A filter, Lempel Ziv compression, encryption and FEC are applied at node A. Their correspondent UNDO parts are applied at node D. In this case, the FEC adapter increases the amount of data sent and over-uses the bandwidth resources of links BC and CD with no benefit. Link BC is especially sensitive to the amount of data transferred because it is dial-up link; transferring unnecessary FEC information can clog it. On the other hand, node A may not have enough execution resources to run four adapters, three of which require major processing of the data.

In both cases the problem of ordering the adapters cannot be solved in any naïve fashion and requires more sophisticated mechanism.

Thus, primitive heuristic approaches cannot provide a sufficient solution to the planning problem.

3.4 Template Planning

Another kind of heuristic planning is template planning [Merigu99] and [Yarvis00]. One simple solution is to precompute a set of reusable plans suitable for common circumstances. The planner needs merely to find a match for its observed problems from among the set of precomputed plans. This solution has the advantage of requiring a very unsophisticated planning component, but the disadvantage of little flexibility. It can only deal with specific sets of problems.

The template planning approach may not be able to find a proper plan for situations that do not fit the available templates. It is hard to imagine that all possible adapters, network problems, and node execution resources can be predefined in the limited number of template scenarios that can be practically created.

Template planning is not flexible. For example, if a new adapter is designed, it requires recalculation of all previously calculated templates to define the relation between the newly designed adapter and previously designed adapters.

Template planning also requires that the templates be calculated in the first place using a real planner, although it presumes that they can be calculated off-line, which could remove a major complaint for ONA planning.

. The basic idea can be extended to allow precomputed plans with slots to be filled in by the planner at runtime [Merigu99], or by allowing the planner some flexibility in location its adapters. The more powerful the extensions, the greater the flexibility, but the greater the complexity of the planner.

3.5 Heuristic Search

As we showed at the beginning of this section, ONA planning is an NP-complete problem. One feasible approach to this problem uses heuristic search in the space of possible solutions. Heuristics provide search hints to the planner that allow it to prune the search tree and get the result in polynomial time. The heuristic search depends on the physical properties of the real environment of networks, adapters, and social relations between ONA-enabled node owners, adapter designers, planner designers, etc. In

Section 3.1 we described the physical model for the heuristic approach that will be considered in this dissertation. The goal of the heuristic planning is to:

- Find an effective plan that solves the problems of a connection.
- Ensure that the plan is feasible, i.e., plan can be deployed on the nodes of the connection.
- Make the search of suitable plans in the space of possible plans fast enough to serve real-time applications.
- Ensure that the chosen adapters are accessible by the networks in a real-time environment.

The drawback of heuristic search is that it depends on the initial state of the search. Some initial states can bring you to a solution that is not necessarily optimal. In the extreme case, heuristic search may not find any feasible solution even if one exists. The art of the planner design is to determine the correct set of heuristics that allows one to find a satisfactory solution in the majority of real-life cases.

In subsequent sections we will demonstrate our approach to heuristic search.

3.6 Routing Around Network Resources

Routing around network resources is a completely different approach to planning. This model is presented in [Choi00]. Networks offer various resources to connections. The planning goal is to find a route in the networks that would satisfy the application requirements of the network channel. If a service located somewhere in the network must be provided to the data flow, the route found by the planner should contain the node

with that service. Performance analysis of this planning approach showed that it has the same complexity as *the graph shortest path problem*.

This method of planning serves a different physical model, where adapters do not move, but the data flow is routed to them. The presumption is that the network has a route that satisfies the application requirements, which may not be the case. The route may require a long detour that makes the latency unacceptable for the application, or may not exist at all. An approach that incurs some short delay for plan calculation and deployment before the connection is established may look more promising. The strong argument for this approach is that if there is a route that avoids the problem and can be found with the same speed as an Internet route, it can be used with less effort.

We think that these two approaches are complimentary. Both approaches have their advantages and disadvantages.

3.7 Automated Planning

Planning can be done manually by a user, or automatically by the user application, or automatically by network software. Proper planning requires knowledge about the properties of an application, network conditions and competence in planning. Thus, automatic planning by the network that also has access to application requirements is the most suitable approach. Access to application requirements can be achieved though interception and analysis of the packets that the application forwards to a destination, or through an API that allows an application (aware of that service) or a user to tell the network what the requirements are. It is likely to be easier for a user or an

application to express the requirements to the network than for a application to collect and store the information about network conditions.

As we outlined in Section 3.1, a planning system (and automated planning systems are not exempt) must handle the following requirements:

- Adapters are consistent so data integrity is preserved.
- Adapters are properly ordered.
- User application requirements are considered in the plan calculation.
- The planning system handles resource management of the system.
- The system is extensible so that the plan and adapter designers can work on their issues autonomously.

Automated planning can be implemented in the different fashions described in this section. In this dissertation, all future discussions of planning will refer to *automated planning*.

3.7.1 Incremental planning

User applications can have particular requirements for how quickly data transfer should start. If the requirement is severe, then ONA nodes may not have enough time for full-scale planning. If data is not sensitive to security at all, or some information spill is not harmful for the application, the data transfer may start immediately no matter how low the QoS is. For example, video-on-demand (VoD) can afford to transfer some frames without encryption, allowing a potential eavesdropper to see some fragments of the video before it is be encrypted and unavailable for him. On the other hand, credit card transactions require full security for every single bit of data. These transactions are

short, and it can be unwise to force them to wait until a full-scale plan is calculated and deployed. Other applications may require a fast start of the data transfer with complete security. Incremental planning is designated to handle these different requirements. Once a central plan is ready, it will replace the incremental plan.

The main disadvantage of centralized planning is that the search through the plan space can be a time-consuming operation that might be unsuccessful. However, the space of the search can be greatly reduced if a plan is calculated *locally*, just for the link where the problem occurs. The consecutive calculation of local plans for each neighboring pair of nodes of the connection will produce an *incremental* plan. Local planning is relatively fast, as it consists of trivial plan-per-link plans. Incremental planning presumes that each node that participates in the communication session is aware of the network conditions on adjacent links, execution capabilities of the neighboring nodes, and available adapters. The node builds a local plan based on knowledge of the up-stream nodes, the up-stream planning data, and the local plans that were already built by up-stream nodes. Only locally available adapters are chosen for local plans. Hence the deployment of a local plan is trivial. Nodes cannot change any decision made upstream.

The main disadvantage to incremental planning is the inefficiency of the overall plan. For example, our communication consists of source node A, destination node C, and intermediate node B (see Figure 3.7). Assume that the user application needs a throughput of 10Mbps. Link A-B has a spare throughput of 5Mbps, and link A-B has a throughput of 2.5Mbps. A must reduce the user data stream by half, from 10Mbps to

5Mbps. It builds the corresponding plan and deploys the adapter that compresses user data by half. B obtains planning data from up-stream. It knows that the user desires 10Mbps, but the A-B link can supply only half of the necessary throughput. B deploys the adapter that decompresses data back to 10Mbps. Then B deploys a compressor that reduces the data by three-quarters. C deploys the adapter that will decompress data back to 10Mbps. As we see, the data was compressed and decompressed twice instead of using only one four-fold compression on A and one four-fold decompression on C. Incremental planning produced an inefficient planning solution. Only a global planning protocol, centralized or distributed, can notice this kind of inefficiency and instruct A and C to run the correct compression.

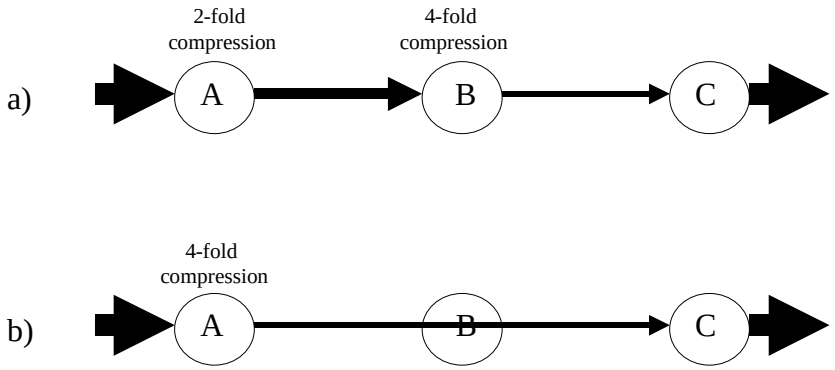

Fig. 3.7: Efficiency of planning: a) inefficient plan b) efficient plan

Combining both incremental and centralized planning can provide of the advantages of both approaches. A communication can use an incremental plan first,

which can be deployed relatively fast, and then switch to a centralized plan whenever it is calculated and deployed.

3.7.2 Central planning

Central planning consists of three important stages:

- Collection of planning data
- Calculation of a plan using the planning data
- Deployment of the plan

A source node initiates collection of the planning data by sending the planning request to the next node in the path to the destination according to ONA routing. The source node puts the application preferences and its local planning data in the request. The connection nodes add their local planning data to the request and forward it further toward the destination. Once the destination receives the request, it forwards it to the site that is designated to run the planner. The planner calculates the plan and forwards it to one or more sites that store the adapters that were chosen for the plan. The adapter storage site runs the deployment protocol, which then delivers the adapters to the connection nodes and instantiates them. Once all the adapters are deployed, the source node starts the data transfer under the newly deployed central plan. The central plan replaces an incremental plan that was calculated before. If the network conditions have changed, *replanning* occurs, and new central plan replaces the old plan.

Calculation of a plan based on user application preferences and complete knowledge about the network should produce the best possible plan. The problem is that planning data collection and plan deployment require network communication and

depend on its cooperation. Plan calculation depends on the sophistication of the planner and the availability of execution resources at the planning node. These factors make the latency of central planning a serious concern for an ONA planning system.

More details of central planning will be presented in Chapter 4.

3.7.3 Distributed planning

Distributed planning uses parallelism in the planning process to improve its agility and reduce its latency. Distributed planning can be implemented using sophisticated negotiation protocols between the planners that participate in the connection. Like most distributed solutions, it is more complex than a central solution. We will consider one possible design of distributed planning.

Let us assume that our connection can be divided into partitions. Each partition runs its own planning process. For the rest of this section, we will use term *partition* planning for our approach.

Partition planning may be an approach if central planning is not possible or constrained by latency. Assume a peer-to-peer communication session between source and destination, through a chain of nodes. The chain is divided into some number of partitions. The planning data does not traverse the links that separate the partitions. Instead, such data is sent to the particular planning site that belongs to the partition. This planning site plans for the partition, updates the current data format status and forwards the planning data to the next partition. The last partition must convert the user data from its current format to the original format. The planning protocol described here is faster

than central planning for two reasons. First, parallelism occurs in two phases of the protocol:

- The collection of planning data occurs faster than for central planning because the partitions are shorter than the whole connection. Hence the plan calculation starts earlier for each partition than it would for the whole connection.
- The deployment of the partition plan starts in closer partitions before data collection and planning are completed in the farther partitions.

Second, partition planning is simpler than planning for the whole connection.

The drawback to this method is that the efficiency of the whole plan is lower than the central plan for the same reasons that the incremental planning is less efficient than central planning. However, distributed planning is more efficient than incremental planning because every partition runs its own central plan.

3.7.4 Where to run central planning

Central plan calculation can occur in different places. It can run on the node that is closest to the connection node where the central data is first collected, for example, on the destination node. Planning can run as close as possible to an adapter storage site to make the latency of the plan delivery low. It can run on the source node in order to monitor other connections that were initiated by the source node. For example, the source node can preempt old connections or redivide its resources among old connections and the new one. Planning can run on any other node, even one that does not belong to the connection, if the node is trustworthy, competent, computationally powerful, cheap

with respect to monetary cost, etc. The source node should be able to choose the site that will run the planning.

3.7.5 Reuse of earlier computed plans

Reuse of earlier computed plans is a very attractive idea for two reasons: the plan is already calculated, and the adapters are already deployed on the connection nodes. The decision about reuse of the plan can be made by a source node or a planning site if the network circumstances and user preferences are the same as before and the previous connection with the plan was successful. The plan should be handed again to the adapter storage site in case it is necessary to reinstall some adapters that had expired on the connection nodes.

3.7.6 Approach to planning for multicast communication

Planning for multicast communication adds another dimension of complexity to the planning problem. It raises a lot of questions that do not have easy answers so far. Adding new users to a multicast tree undermines the whole idea of an optimal plan. Consider the example in Figure 3.8. Assume that we have a multicast tree with users C and D that receives a video stream from node A node B. Link AB, adjacent to the source node, is low bandwidth. All receivers prefer a low-resolution color movie to other feasible alternatives, so the planner puts a quality-dropping filter on node A. If another node E that prefers a black and white high-resolution movie joins the multicast tree, not many things can be done to satisfy the connection:

- The new user preferences can be ignored.

- Another black and white high-resolution stream can be generated that will share link AB with the original colored low-resolution stream, making link AB even worse (Figure 3.8a).
- Another black and white high-resolution stream can be generated that will use an alternate route AE (Figure 3.8b).

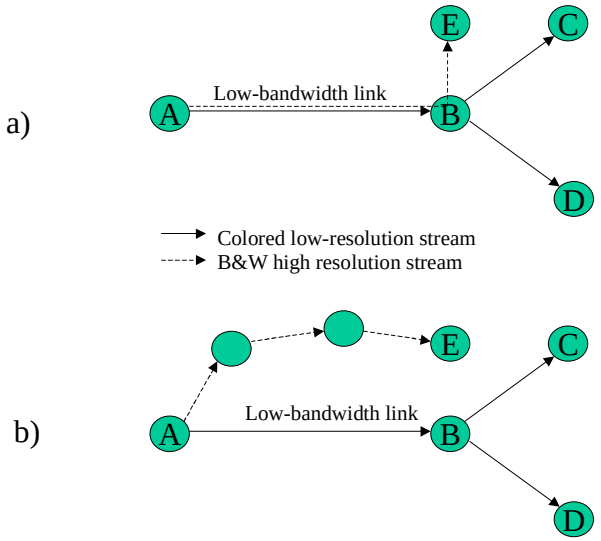

Figure 3.8: Multicast tree planning

Only the last option can satisfy all users, but it requires recomputing the tree topology every time a new user joins the session. Also, it presumes a alternative route is available. If a new user with specific preferences joins the tree, the planner must find a point on the tree where the user preferences can be satisfied. The planner should also find a distinct route from the new user to that point, instead of just using the closest point that belongs to the multicast tree. User preferences are not only the reason for such situations.

Consider another example of the complexity of multicast tree planning. Assume that we have a multicast tree where users listen to an encrypted stream (as shown on Figure 3.9). Node E joins the tree through node B, but link BE is insecure and of low bandwidth. To make the connection BE feasible, compression should be added. Because the stream is encrypted, in order to apply compression, it should first be decrypted, than compressed and encrypted again. The compression can be added in two different ways:

- Decryption, compression, and encryption is added to node B
- Compression is added to node A, decompression is added to nodes C and D

In the first case, the replanning does not affect the up-stream tree, but the plan is not optimal. Nodes C and D can be affected by the new plan because node B must run extra adapters. In the second case, replanning of the whole tree may be required. Nodes C and D will have to run decompression that earlier they did not have to run. Hence, in all cases, if any node joins the connection it affects a big part of the tree that can require replanning and redeployment at large scale.

The calculation of the multicast tree with a stable topology is a very complicated problem itself because each position of an adapter can affect many subsequent tree brunches.

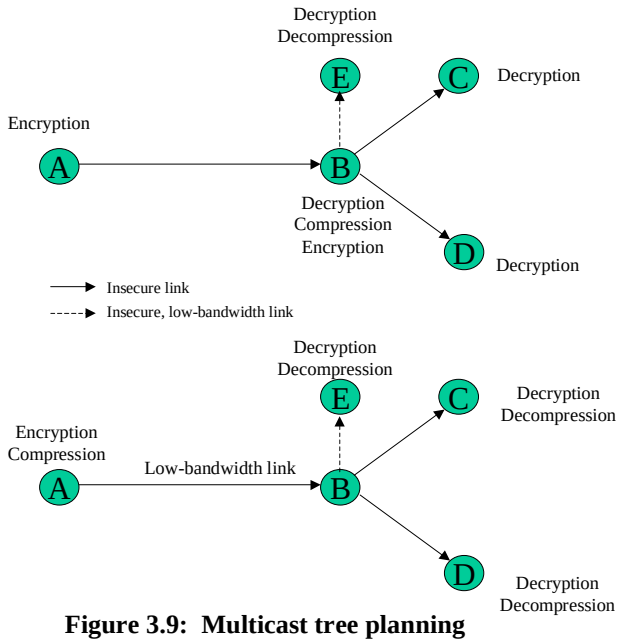

Figure 3.9: Multicast tree planning

The problem of multicast tree planning is not well explored yet; we will return to it in Chapter 8, Future Work.

3.8 Adapter Design

The design of adapters is a very complicated problem because they should be used together with other adapters, in different circumstances, by a planner that was designed by another party, or that was designed later than the adapters were designed. This problem requires proper classification of adapters and proper standards of the description of adapters and their properties, the effects that they provide, and the resources that they require. These requirements are briefly described in this section.

3.8.1 Classes of adapters

Adapters can be lossy (if their activity causes the loss of user data) or non-lossy. Adapters can be unary if they contain only one part (for example, filter and format converters) or binary if they contain DO and UNDO parts (such as compressor/ decompressor).

We presume that the possible universe of all adapters can be reduced to a limited list of adapter classes where each class is associated with a particular network problem. Each class can be divided into subclasses; each subclass corresponds to a particular method used for the resolution of the problem. For example, the data reduction class consists of the *lossy adapter* subclass and the *compressor* subclass. In the future, new network problems will be recognized and new methods of their resolution will be invented, but we do not expect that those numbers will grow fast.

We distinguish the following main classes of adapters:

1. Data storage
    - Buffering
    - Caching
    - Prefetching
    - Scheduling or prioritization
2. Data Reduction
    - Lossy adapters (filters frames, sound, fragments of images, etc.)
    - Lemple Ziv type compression
3. Data Encryption

- Encryption
- Padding (adding noise to camouflage real data)

4. Forwarded error correction
5. Authentication
6. Format conversion
7. Network enhancement
    - Avoidance of congestion
    - Search for an alternative path
    - Multipath data delivery
    - Network sensing
    - Camouflaging real network traffic.

Adapters are classified by the methods they use and the effects they provide. For example, we are interested how much data is compressed, or whether applying a particular adapter affects compressability of data, i.e. the ability of the data to be compressed. There are as many compressabilities as there are ways to compress the data. For example, the color compressability shows whether it is possible to compress the data by taking away the color. Resolution-drop compressability describes whether it is possible to drop the image resolution to a particular level. Adapters that use different methods can be used together for a stronger effect. For example, a filter that drops color can be used together with a resolution-dropping filter. The dropped data cannot be restored to full extent. The filters are data type and format sensitive. Thus, color or

image-resolution can be dropped from images only. If an image is compressed or encrypted, the filtering may not be possible.

Some adapters are sensitive to the format of the data, and this information is critical for the adapter's use. Every adapter uses the particular execution resources of its node; the designer of the adapter must provide this information if the planner is to work accurately. If an adapter designer fails to provide this data or the data provided is not correct or true, the adapter cannot be used because its effects might be disastrous. The presumption is that any adapter designer is interested in making his adapters widely used, and the origin of every adapter is traceable no matter how far it travels.

3.8.2 Adapters with flexible/rigid architecture

Adapters can have flexible or rigid architectures. Flexible adapters can change their effects during the data transfer if the conditions of the connection change. They can avoid the need to replan and redeploy adapters. But they require a high level of sophistication in their design and in the planning model. Most adapters have a simple, rigid architecture, which does not allow changes to adapter properties during a data transfer. The use of these adapters requires replanning and redeployment of adapters in the case of a change in connection conditions, but it keeps the planning model simple.

3.8.3 Two-level hierarchy of access to adapters

Information about adapters must be easily accessed by the planner. It is provided in a two-level structure. The first level is the planner database, where all known network problems are associated with adapter packages that are assigned to solve them. The

adapter packages are supplied by adapter designers. The records about the packages must be added to the planner database. In these records, each problem is associated with one or more of the adapter packages designated to solve them. The records also contain the location of these packages and the interface used to access these adapter packages. Thus, from the planner database the planner can find the adapter package that resolves a particular network problem and an interface to the correspondent adapter package.

Each package has its own database that contains the necessary description of the adapters that belong to this package. This database is created together with the adapter package by the same adapter designer. The database contains the records that completely describe adapters. These records contain data about the methods of adapters, the effects of adapters, and the resource costs of the adapters. It also contains the names of adapters and how to use them. A package database can be any kind of database: relational, object-oriented, or even a text file. If the whole adapter package contains only one adapter, it should have a reference that plays the role of the database. How to access these databases should be properly described in the planner database.

The planner chooses a package of adapters, queries the package database, and obtains the name of an adapter, the description of the adapter, and its location. It uses this data for the plan calculation. The adapters might not be physically stored on the same node as the planner, but the package database should point to their remote location. However, most of the time the adapters and the planner with its two-level database are located on the same node.

3.9 Summary

We considered a number of problems inherent in planning for ONA. Some of these problem-solving approaches are implemented in our system, as described in the next chapters. For some other approaches, we have offered design suggestions. Some of these approaches are addressed in the chapter on future work.