Service Oriented Resource Orchestration in Future Optical Networks

Chinwe E. Abosi, Reza Nejabati, Dimitra Simeonidou
School of Computer Science and Electronic Engineering
University of Essex, Colchester, UK, CO4 3SQ
Email: {ceabos, rnejab, dsimeo}@essex.ac.uk

***Abstract*—The future Internet evolution is driven by applications that require simultaneous real-time access to multiple heterogeneous IT resources interconnected by high-speed optical networks.**

In this paper, we propose a novel service-oriented resource orchestration model based on the optimization of heterogeneous IT and network resources owned by different Infrastructure Providers (InPs). The proposed model aims to achieve a global optimum from both the end-users' and InPs' point of view across different administrative domains. An integer linear program (ILP) is formulated to obtain optimal results that maximizes the number of accepted requests while minimising resource usage. It is compared against a co-scheduling ILP model, whose objective is to maximise the number of accepted requests only. Finally, we propose a heuristic solution for scalability. Its performance is compared against (i) our proposed ILP (ii) a co-scheduling heuristic that aim to maximise number of accepted requests only and (iii) an algorithm that does not take into account cross-domain optimisations.

I. Introduction

The distributed and resource intensive nature of emerging Internet applications necessitates the need for a scalable, yet efficient coordination of disparate resources across heterogeneous Infrastructure Providers (InPs). These applications require the transfer of huge amounts of data across the network to remote sites for processing, storage or content delivery. Furthermore, they may require guaranteed levels of Quality of Service (QoS). Optical Wavelength Division Multiplexing (WDM) networks are the only transport medium that can support such high-bandwidth-intensive applications, and offer end-to-end QoS-guaranteed network bandwidth services needed to perform the required data transfer. However, the heterogeneous nature of the underlying resource infrastructure (i.e. computational vs. storage vs. network bandwidth) plus the competitive relationship that may exist between InPs can make providing the type of end-to-end services needed by these emerging applications a daunting task. To address this, we proposed a Service Oriented Architecture for the Future Internet (SOAFI) framework shown in Fig. 1 capable of organising resources from different administrative domains to provide enriched future Internet services [1].

Service Oriented Architecture, as defined by The Organization for the Advancement of Structured Information Standards (OASIS) [2], is a *"paradigm for organizing and utilizing distributed capabilities that may be under the control of different ownership domains ... to produce desired effects consistent with measurable preconditions and expectations."* The SOAFI framework provides an abstraction mechanism [1], [3] through which information from different InPs can be described and disseminated onto a Service Plane (SP). The SP [1] aims to seamlessly coordinate the interaction-between and access-to both network and IT resources in order to facilitate a better and more efficient service provisioning at the individual domains of each InP. Thus it (i) acts as a "matchmaker" between users and the multiple InPs that may have the resources they require and (ii) creates a collaborative environment that facilitates the interaction between both network and IT resources and the InPs that provide, control and manage these resources.

Resource orchestration refers to the locating, coordinating and selecting of resources from various InPs to fulfil a users requirement. It aims to select the 'best' or most optimal IT resource site and network path to this selected IT resource site to satisfy a job request. The selected resource information is sent to the control and management entities of the respective InPs (the local resource management systems (LRMS) of IT resources and Network Resource Provisioning System (NRPS) of the optical network) through a standard interface. InPs then use this information to perform low-level task scheduling and lightpath and wavelength assignment accordingly.

In this paper, we implement the proposed service-oriented resource orchestration algorithms. The aim of the algorithm is to select appropriate resources from multiple InPs to improve service provisioning in two ways - (i) maximising accepted requests while (ii) minimising resource usage in terms of number of hops. The algorithm ensures that when the task scheduling and routing and wavelength assignment is implemented on the selected resources by the individual underlying infrastructure's control and management systems, a global optimum across all the domains can be achieved.

We formulate and present the resource orchestration problem as an Integer Linear Programming (ILP) model. We compare the proposed ILP with an ILP formulation that aims to maximise the number of accepted requests. A near-optimal heuristic approximation algorithm is also presented for scalability and compared to the ILP formulation over a small 5-node network. The heuristic is then studied over the larger European Optical Network (EON) and the NSF network and compared against an algorithm that attempts to sequentially locate and select resources from individual InP domains.

978-1-4577-0638-7 /11/$26.00 ©2011 Crown

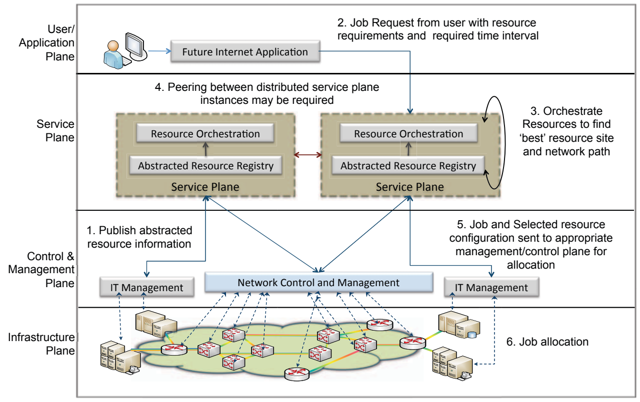


Fig. 1. Architectural Overview

II. Related Works

The closest research topic to resource orchestration as implemented in the SP is the joint or co-scheduling of resources. This involves the scheduling of network and IT resources simultaneously to support distributed applications. It normally involves a broker, which exists within either the middleware, management or control planes, tasked with joint scheduling of resources. The broker also implements low level task scheduling and routing and wavelength assignment [4], [5] to select, schedule and allocate required resources.

Meta-Scheduling Systems (MSS) that implement joint scheduling of resources, such as the Data Intensive And Network Aware (DIANA) environment introduced in [6] and the Joint Resource Scheduling System (JRSS) introduced in [7], are closely related to the proposed SP resource orchestration. Like the SP, the authors in [7] present a joint meta-scheduling system that co-allocates network and IT resources in a separate, yet integrated entity. These meta-schedulers use co-scheduling algorithms [8]–[11] to jointly select network and IT resources. [8] aims to maximise the number of accepted request while minimising an arbitrary cost based on the requested capacity to execute the request. [9] considers the maximisation of the economic benefit to InPs by introducing an economic cost for using a resource and a budget a client is prepared to pay for accessing the resource. [10] presents two separate objective functions to maximise the number of accepted request and minimise the number of wavelengths used. The authors in [11] propose a joint scheduling model which first sorts the jobs to be scheduled into a list according to some defined priority of each job and then schedules each job to a specific resource.

Like MSSs, the SP facilitates the requesting of resources from more than one InP, and schedules the required resources across these InPs. Unlike the DIANA and JRSS MSS proposals, which implements resource management, low-level task assignment and application execution; resource orchestration performs a high-level co-scheduling of resources using abstracted resource information. A site and path is selected by the resource orchestration algorithm, while the actual allocation of the job on individual processors, diskspace locations or wavelengths is left to be implemented by the LRMS and NRPS of the underlying infrastructure.

III. System Model

The infrastructure modelled includes (i) IT resource nodes, $\mathcal{R}$; and (ii) users that generate job requests, interconnected to (iii) an optical mesh network. IT resource sites offer either data storage resources, $S \subseteq \mathcal{R}$, computational resources, $C \subseteq \mathcal{R}$, or video servers, $\Phi \subseteq \mathcal{R}$, containing files that need to be streamed to the requester. Each compute site, $c \in C$ is described by its residual processing capacity, P_c, in MIPS. Each storage sites, $s \in S$, has a finite capacity represented by D_s. Each video server, $\phi \in \Phi$, can serve a maximum number of concurrent sessions. IT resource sites, $\mathcal{R}$, and users are connected to the network through opaque OXCs with optical bypass. The network is an optical circuit switched network consisting of N nodes, $\mathcal{V}$, connected by optical fibre links, $\mathcal{E}$. Each link, $e_{(u,v)}$ has W wavelengths, each with a finite transmission capacity $B^w_{(u,v)}$. We assume that each OXC is equipped with unlimited wavelength conversion and traffic grooming capabilities. In this way, a lightpath can be established between any source-destination pair as long as there is enough bandwidth resources on the links in the path. Video servers are assumed to have the requested files, allow unlimited concurrent sessions, and satisfy all media-related constraints, hence the only constraints we consider for these requests is link bandwidth. To simplify the problem, we assume that the links connecting IT resources and users to the network have infinite bandwidth. Hence we can neglect this connection in our model.

There are two types of job requests, $j \in J$. Type A requests are either computational, $J^{A_{comp}}$ or storage $J^{A_{stor}}$ requests. They are associated with a 5-uple $(src_j, b_j, r_j, t_j^{start}, t_j^{end})$ where $src_j \in V$ is the source; b_j is the required bandwidth. For computational requests, data is sent on output links (ε^{out}) to a computation site, processed at the site and the resulting data is returned on input links (ε^{in}) to the requester. For storage requests, data is sent on output links (ε^{out}) to a storage site. r_j is the requested IT resource capacity equal to p_j if the requested resource type is computational and d_j for storage requests. $[t_j^{start}, t_j^{end}]$ is the time interval during which the job is active. Type B requests are video requests, represented by a 5-uple $(src_j, \phi_j, b_j, t_j^{start}, t_j^{end})$, where $src_j \in V$ is the requesting node; $\phi_j \in \mathcal{R}$ is the id of the streaming server node. For this request type, bandwidth is scheduled on input links (ε^{in}) to mimic the streaming of files from the server to the requester. b_j and $[t_j^{start}, t_j^{end}]$ are as previously defined.

IV. Resource Orchestration in the Service Plane

In this section, we introduce an Integer Linear Programming, ILP, formulation for the resource orchestration problem and an approximation heuristic for scalable evaluation.

A. Integer Linear Problem Formulation

To formalise the ILP, we introduce the following notations and binary variables:

- θ_j is a binary variable, $\theta_j \in [0,1]$, where θ_j = 1 if job j is accepted. θ_j = 0 otherwise.
- $\varepsilon^{in/out}_{j,(u,v),t}$ is a binary variable, $\varepsilon^{in/out}_{j,(u,v),t} \in [0,1]$, where $\varepsilon^{in/out}_{j,(u,v),t}$= 1 if job j uses link (u,v) for input/output data at time instance, $\tau_t \in [t_{start}, t_{end}]$. $\varepsilon^{in/out}_{j,(u,v),t}$ = 0 otherwise.
- $\gamma_{j,c,t}$ is a binary variable, $\gamma_{j,c,t} \in [0,1]$. $\gamma_{j,c,t}$ = 1 if job j uses a compute node at an IT resource site c for processing data at time instance, $\tau_t \in [t_{start}, t_{end}]$. $\gamma_{j,c,t}$ = 0 otherwise.
- $\delta_{j,s,t}$ is a binary variable, $\delta_{j,s,t} \in [0,1]$. $\delta_{j,s,t}$ = 1 if job j uses a storage node at an IT resource site s for storing data at time instance, $\tau_t \in [t_{start}, t_{end}]$. $\delta_{j,s,t}$ = 0 otherwise.

Our objective, given in Equation 1, is to ensure that the underlying network resources are used efficiently in satisfying job requests. We consider only network resources in this objective function. This is because a single job request can use more than one network link to satisfy its requirement. In this case, the requested bandwidth is replicated on each of the links. However, each job request can be satisfied on only a single IT resource or none at all. Thus our system model does not allow for the possibility of resource wastage on IT resources which can occur on network resources.

$$RU = \min \sum_{e_{(u,v)} \in \mathcal{E}, j \in J} \varepsilon_{j,(u,v)} \tag{1}$$

The set of constraints is the following: The first constraints, Equations (2) and (3), ensure that the maximum number of requests can be accepted.

$$\sum_{j \in J} \theta_j \geq ACC \tag{2}$$

where ACC represents the maximum number of requests that can be accepted. If there is sufficient capacity to accept all the job requests, Equation (2) transforms to:

$$\sum_{j \in J} \theta_j \geq |J| \tag{3}$$

$$\sum_{j} p_j \cdot \gamma_{j,c,t} \leq P_c, \quad \forall c \in C \; \forall t \in T \tag{4}$$

$$\sum_{j} d_j \cdot \delta_{j,s,t} \leq D_s, \quad \forall s \in S \; \forall t \in T \tag{5}$$

$$\sum_{j} b_j \cdot \varepsilon^{in}_{j,(u,v),t} \leq B^w_{(u,v)} \cdot W, \quad \forall e_{(u,v)} \in E \; \forall t \in T \tag{6}$$

$$\sum_{j} b_j \cdot \varepsilon^{out}_{j,(u,v),t} \leq B^w_{(u,v)} \cdot W, \quad \forall e_{(u,v)} \in E \; \forall t \in T \tag{7}$$

Equations (4) - (7) are capacity constraints. They state that any job request that is to be satisfied at processing site, c, storage site, s, and/or link, $e_{(u,v)}$ at time t must not exceed the maximum capacities available for each type of resource.

$$\sum_{c} \gamma_{j,c} = \theta_j, \quad \forall j \in J^A_{comp} \tag{8}$$

$$\sum_{s} \delta_{j,s} = \theta_j, \quad \forall j \in J^A_{stor} \tag{9}$$

$$\varepsilon^{in}_{j,(u,v)} \leq \theta_j, \quad \forall j \in J \; \forall e_{(u,v)} \in E \tag{10}$$

$$\varepsilon^{out}_{j,(u,v)} \leq \theta_j, \quad \forall j \in J \; \forall e_{(u,v)} \in E \tag{11}$$

Equations (8) - (11) are the resource allocation constraints. (8) and (9) force each storage or computational job request to use only one storage site or one computational site respectively based on its requirement. Bandwidth allocation is forced by (10) and (11) to use as many links as needed for each accepted job request.

$$\forall j \in J \; \forall u \in V$$

$$\sum_{\substack{v \in V \\ e_{(u,v)} \in E}} \varepsilon^{in}_{j,(u,v)} - \sum_{\substack{v \in V \\ e_{(v,u)} \in E}} \varepsilon^{in}_{j,(v,u)} = \begin{cases} +\theta_j & \text{if } u \mapsto \Phi \\ +\gamma_{j,u} & \text{if } u \mapsto C \\ -\theta_j & \text{if } u \mapsto C, \Phi \\ 0 & \text{otherwise} \end{cases} \tag{12}$$

$$\forall j \in J \; \forall u \in V$$

$$\sum_{\substack{v \in V \\ e_{(u,v)} \in E}} \varepsilon^{out}_{j,(u,v)} - \sum_{\substack{v \in V \\ e_{(v,u)} \in E}} \varepsilon^{out}_{j,(v,u)} = \begin{cases} +\theta_j & \text{if } u \mapsto C, S \\ -\gamma_{j,u} & \text{if } u \mapsto C \\ -\delta_{j,u} & \text{if } u \mapsto S \\ 0 & \text{otherwise} \end{cases} \tag{13}$$

The flow conservation for the network resources are forced by Equations (12) and (13). The equations ensure that the correct direction of flow for each job request is as detailed in Section III. Furthermore, the equations also ensure that if a job, j is accepted, then all required resource types for job j are allocated or none of the resource type is allocated. The right side of both equations ensure that the correct type of resource is used to satisfy a requirement. The notations, $u \mapsto C, u \mapsto S, u \mapsto \Phi$, associate compute and storage resources as well as servers with the edge nodes of the network.

$$\varepsilon^{in}_{j,(u,v)} + \varepsilon^{in}_{j,(v,u)} \leq 1 \qquad \forall j \in J \; \forall e_{(u,v)} \in \mathcal{E} \tag{14}$$

$$\varepsilon^{out}_{j,(u,v)} + \varepsilon^{out}_{j,(v,u)} \leq 1 \qquad \forall j \in J \; \forall e_{(u,v)} \in \mathcal{E} \tag{15}$$

Equations (14) and (15) avoid loop formation by ensuring that a link, $e_{(u,v)}$ is not used in both directions to satisfy a job request.

B. *Offline Heuristics*

We introduce a Modified Simulated Annealing (MSA) heuristic for the orchestration of resources since ILPs cannot solve large complex problems in realistic times. Our objective is to minimise the number of hops used in accepting the maximum number of job requests. The MSA algorithm is based on the Simulated Annealing (SA) algorithm which is a generic probabilistic meta-algorithm used to solve global optimization problems [12]. The pseudo-code of the MSA algorithm is shown in Algorithm 1. It starts off with an **initial solution** that has an initial **cost**. It then runs through a number of iterations at a given temperature, during which

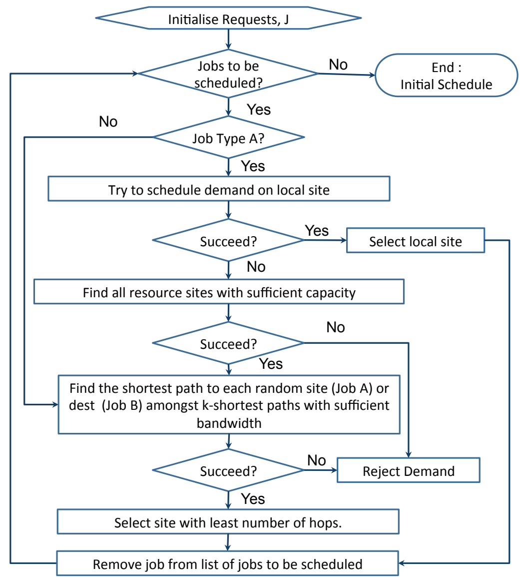


Fig. 2. Flowchart to Construct MSA Initial Solution

a new solution and its corresponding cost is obtained by a **perturbation** scheme. Depending on the cost, the solution from the perturbation is accepted or rejected.

The **cost** of a solution in the MSA is defined by the total number of hops used in satisfying a maximum number of requests. The flow-chart to create the **initial solution** is shown in Figure 2. For job-type A, the algorithm selects the closest IT resource site. Thus, it first attempts to use the IT resource site connected to the same edge node as the client node. If such an IT site exists and it has sufficient capacity, the job request is scheduled on that site. If such a site does not exist, the algorithm selects the closest IT resource site with sufficient resource capacity. It then routes the job request on the shortest path, $\kappa \in K, 1 \leq \kappa \leq K$, with sufficient bandwidth capacity. If a $(c, \kappa^{fwd}, \kappa^{rev})$ triple for $j^{A_{comp}}$ request or (s, κ^{fwd}) tuple for $j^{A_{stor}}$ request with sufficient capacity cannot be found, the request is rejected. This approach aims to create an initial solution using a minimum number of hops. For job-type B, the request is routed on the shortest path with sufficient bandwidth capacity among k-shortest paths, $\kappa \in K, 1 \leq \kappa \leq K$. If no such path is found, the request is rejected.

The **perturbation** process generates new solutions in the neighbourhood of the current solution. It proceeds by freeing a random number of job requests and then attempting to re-orchestrate all unorchestrated job requests on a random IT resource site and a random path among the k-shortest paths. The change in cost, (Δ), between the new solution and the current solution is calculated. The solution is accepted or rejected according to the Boltzmann probability given in Equation (16).

$$P_{boltz}(\Delta) = \exp(\frac{-\Delta}{T}) \tag{16}$$

where P_{boltz} is the Boltzmann probability of accepting the new solution, T is a control parameter known as the current "temperature" and $-\Delta$ is the improvement to the solution.

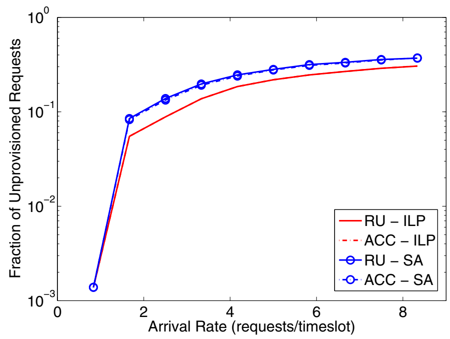
Algorithm 1 Modified Simulated Annealing

$\{J\}$ set of requests to be scheduled
Initialize: $T_{min} = 0.01$, $T_{max} = 300$, $T \leftarrow T_{max}$
Initialize: cooling factor, $c = 0.88$
Initialize: $currSoln \leftarrow$ Initial Solution using to flow-chart
set of unscheduled requests, $availDemands \leftarrow J$
while $T > T_{min}$ **do**
  $newSoln \leftarrow$ perturb $currSoln$
  $\Delta_1 \leftarrow newSoln.cost_1 - currSoln.cost_1$
  Generate $u_1 \in [0, 1]$ uniform random variable
  if $\Delta_1 < 0$ **then**
    $currSoln \leftarrow newSoln$
  else if $\Delta_1 == 0$ **then**
    Generate $u_2 \in [0, 1]$ uniform random variable
    $\Delta_2 \leftarrow newSoln.cost_2 - currSoln.cost_2$
    if $\Delta_2 > 0$ or $u_2 < P(\Delta_2)$ **then**
      $currSoln \leftarrow newSoln$
    end if
  else if $u_1 < P(\Delta_1)$ **then**
    $currSoln \leftarrow newSoln$
  end if
  $T = T_{max} * c$
end while
return $Solution$

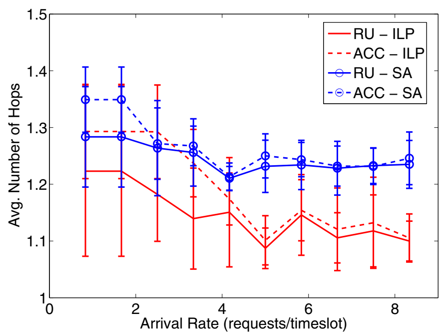
V. Validation and Analysis

For our analytical model and simulation, we consider a 5-node topology to evaluate our ILP and to validate the heuristic solution. We then do a scalable analysis using the NSFNET and the European Optical Network (EON).

We first evaluate our ILP and use it to validate the heuristic over a 5-node topology connecting four resource nodes for tractability. The link bandwidth is set to 40 Gbps, with 8 wavelengths per link. The resource capacity of each storage/computational node is randomly selected from $\{1250, 1500\}$. The request set used in this analysis is created randomly. We recall that Job Type A requests are storage or computational requests and Job Type B requests are video requests. The start time is uniformly distributed between $[0, 22]$ for Job Type B requests and $[0, 15]$ for Job Type A requests. The duration of each job request is uniformly distributed between $[2, 5]$ for Job Type B and $[5, 23]$ for Job-Type A requests. Request for storage/computational resources is uniformly distributed between $[10, 50]$. Bandwidth requirements are uniformly distributed between $[50, 100]$ Gbps for Job Type B and $[100, 150]$ Gbps for Job Type A. The granularity of each wavelength is 1 Gbps. Thus bandwidth can only be assigned in discrete quantities of 1 Gbps. The number of alternate paths is set to $K = 3$. We vary the number of job requests we simulate between 20 and 200 requests, over 24 timeslots and the results are averaged over 20 simulation runs for which we calculate a 95% confidence interval. We use an Intel Core $i7$, 2.93 GHz machine, with 8 GB of RAM.



(a) Fraction of Unprovisioned Requests



(b) Number of Hops

Fig. 3. Comparing two Objective Functions over the 5-Node Topology

Fig. 3 shows the ILP and heuristic solution for both the RU and the ACC objectives. Recall that RU is our proposed ILP and ACC is an ILP whose objective is to maximise the number of accepted requests [10]. To implement ACC, we take into consideration only relevant constraints from [10] and extend it to introduce video streaming, storage requests, the corresponding resources and relevant constraints.

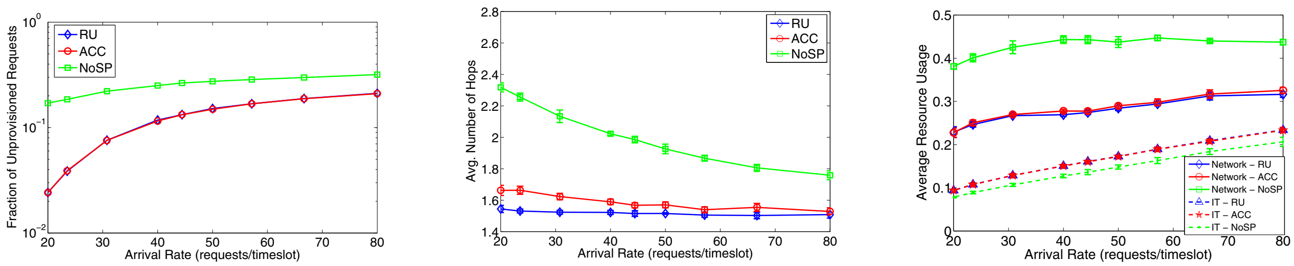
We first measure the performance of our proposed service-oriented resource orchestration ILP, RU-ILP, by comparing it to the ACC-ILP. Fig.3(a) shows the fraction of unprovisioned requests measured against the arrival rate which is the number of requests per timeslot. We observe that RU returns the same results when compared to ACC in terms of rejection ratio. Fig. 3(b) shows that RU outperforms ACC in terms of average number of hops, using 10% less hops on average than ACC, with the least hops used at lower arrival rates.

Next, we validate the proposed heuristic, RU-SA, by comparing it to the ILP formulation, RU-ILP which returns optimal solutions. We observe that our heuristic solution is comparable with the optimal ILP solution in terms of the fraction of unprovisioned requests and number of hops with a 5% difference at the highest arrival rate in both cases.

Finally, we evaluate the performance of our proposed resource orchestration heuristic, RU against two other heuristics as benchmarks: (i) the ACC heuristic, which was validated against the ACC-ILP in Fig. 3 and (ii) an approach which assumes no cross-domain interactions and no cooperation between InPs. We call this the NoSP. In the NoSP, a request is made to each InP one at a time until the complete end-to-end service can be achieved. For scalability, we perform the evaluation over the 32 node EON with 92 unidirectional links, 9 storage resource sites and 9 computational sites and the NSFNET with 14 nodes, 42 unidirectional links, 5 storage resource sites and 5 computational sites. Each link has a bandwidth of 40 Gbps and 16 wavelengths. The storage and computational capacities of each edge nodes is chosen randomly from $[2000, 7000]$ units with a step size of 1000 units. The start time is uniformly distributed between $[0, X-5]$ for Job Type B requests and $[0, X-10]$ for Job Type A requests. X represents the maximum number of timeslots which ranges from $[20, 100]$. The number of requests during these timeslots is 2000, thus the arrival rate (requests/timeslot) is varied. The duration of each job request is uniformly distributed between $[2, 5]$ for Job Type B and $[4, 12]$ for Job-Type A requests. Request for storage/computational resources is uniformly distributed between $[50, 200]$. Bandwidth requirements are uniformly distributed between $[2, 5]$ Gbps for video requests and $[10, 50]$ Gbps for storage/computational requests. The source and the destination for video requests are randomly selected.

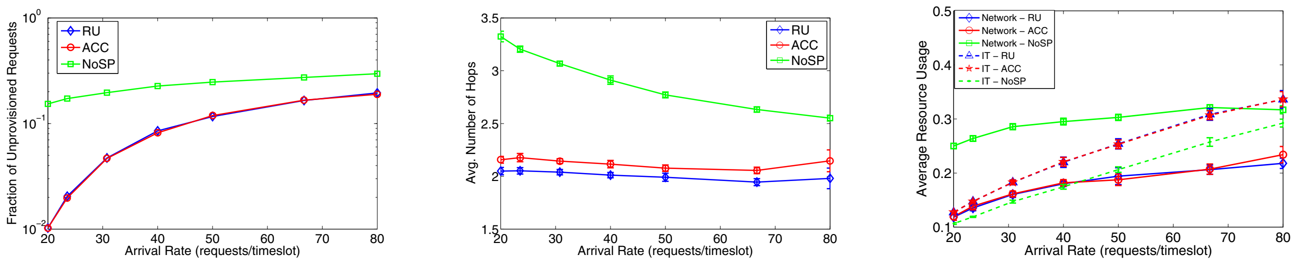
We first compare RU and ACC over the larger NSF and EON topologies. Fig. 4 and 5 show that on comparison, ACC and RU give the same solution in terms of fraction of unprovisioned requests, but RU outperforms ACC in terms of average number of hops and average resource usage. We observe an improvement in network usage, but negligible improvement in IT resource usage. This is due to the fact that RU uses less links on average, thus uses the network resources more efficiently than ACC.

Finally, we compare our resource orchestration (RU) heuristic with the NoSP approach where there is no cooperation between InPs. For the fraction of unprovisioned requests shown in Fig. 4(a) and 5(a), we observe that when resource orchestration is implemented over the EON and NSFNET as opposed to the NoSP, there is 14% and 15% more accepted requests respectively on average. There is also an improvement of 24% and 30% in the average number of hops used on average over the NSFNET (Fig. 4(b)) and EON (Fig. 5(b)) respectively when the RU is used as opposed to the NoSP. From Fig. 4(c) and 5(c), we observe an average improvement of 37% in the network resource usage and 17% improvement in the IT resource usage on the NSFNET. For the EON, we observe an improved usage of 22% and 41% for the network and IT resources respectively. From our result, we observe that our RU approach performs much better in terms of fraction of unprovisioned requests, number of hops and average resource usage than the NoSP approach. This can be attributed to the collaboration that exists when using the resource orchestration of the SP. As the SP is able to globally optimise the selection of resources from amongst the multiple InPs that may be able to satisfy a users requirement in such a way that the resources are optimally utilised, it results in an increase in the number of accepted requests.



(a) Fraction of Unprovisioned Requests (b) Average Number of Hops (c) Average Resource Usage

Fig. 4. Evaluation of Resource Orchestration over the NSFNET Topology



(a) Fraction of Unprovisioned Requests (b) Average Number of Hops (c) Average Resource Usage

Fig. 5. Evaluation of Resource Orchestration over the EON Topology

VI. Conclusion

In this paper, we proposed a service oriented resource orchestration mechanism to coordinate resources from different InPs facilitated by a unified platform, the Service Plane, over an optical network. We developed a new mathematical model for the service-oriented resource orchestration problem and presented a heuristic approximation based on a modified Simulated Annealing heuristic for efficiency and scalability. Our results show that our proposal is comparable to previous studies related to co-scheduling of resource. Moreover our solution outperforms these in terms of resource usage. Our ILP, which returns optimal results, is used to benchmark our heuristic approximation. A scalability analysis of our heuristic approximation is carried out by considering the European Optical Network and the NSFNET to satisfy a larger number of requests. The results shows that the proposed SP resource orchestration performs much better than when there is no orchestration of resources amongst multiple InPs.

Acknowledgment

The work described in this paper was carried out with the support of the GEYSERS (Generalized Architecture for Dynamic Infrastructure Services), and the Mantychore projects funded by the European Commission through the 7th ICT Framework Programme.

References

[1] C. Abosi, R. Nejabati, and D. Simeonidou, “A novel service composition mechanism for future optical internet,” *Optical Communications and Networking, IEEE/OSA Journal of*, vol. 1, pp. A106–A120, July 2009.

[2] (2006, October) Reference model for service oriented architecture 1.0. OASIS Standard. [Online]. Available: http://docs.oasis-open.org/soa-rm/v1.0/soa-rm.html

[3] C. Abosi, R. Nejabati, and D. Simeonidou, “A service plane architecture for future optical internet,” in *Optical Network Design and Modeling, 2009. ONDM 2009. International Conference on*, 2009, pp. 1 –6.

[4] Z. Sun, W. Guo, Z. Wang, Y. Jin, W. Sun, W. Hu, and C. Qiao, “Scheduling algorithm for workflow-based applications in optical grid,” *J. Lightwave Technol.*, vol. 26, no. 17, pp. 3011–3020, 2008. [Online]. Available: http://jlt.osa.org/abstract.cfm?URI=JLT-26-17-3011

[5] X. Liu, C. Qiao, W. Wei, X. Yu, T. Wang, W. Hu, W. Guo, and M.-Y. Wu, “Task scheduling and lightpath establishment in optical grids,” *J. Lightwave Technol.*, vol. 27, no. 12, pp. 1796–1805, 2009. [Online]. Available: http://jlt.osa.org/abstract.cfm?URI=JLT-27-12-1796

[6] R. McClatchey, A. Anjum, H. Stockinger, A. Ali, I. Willers, and M. Thomas, “Data intensive and network aware (diana) grid scheduling,” *Journal of Grid Computing*, vol. 5, pp. 43–64, 2007, 10.1007/s10723-006-9059-z.

[7] W. Guo, W. Sun, Y. Jin, W. Hu, and C. Qiao, “Demonstration of joint resource scheduling in an optical network integrated computing environment [topics in optical communications],” *Communications Magazine, IEEE*, vol. 48, no. 5, pp. 76 –83, may. 2010.

[8] V. T. De, B. Volckaert, P. Thysebaert, M. D. Leenheer, F. D. Turck, B. Dhoedt, and P. Demeester, “Network aware scheduling in grids,” in *NOC ’04: Proc. 9th European Conference on Networks and Optical Communications*, Eindhoven, The Netherlands, 2004.

[9] R. Aoun, M. Gagnaire, “Service differentiation based on flexible time constraints in market-oriented grids,” accepted for publication at GLOBECOM. HNL, USA (2009).

[10] H.-H. Nguyen, M. Gurusamy, and L. Zhou, “Provisioning lightpaths and computing resources for location-transparent scheduled grid demands,” in *Optical Network Design and Modeling, 2008. ONDM 2008. International Conference on*. IEEE, 12-14 2008, pp. 1 –6.

[11] Y. Wang, Y. Jin, W. Guo, W. Sun, W. Hu, and M.-Y. Wu, “Joint scheduling for optical grid applications,” *J. Opt. Netw.*, vol. 6, no. 3, pp. 304–318, Mar 2007.

[12] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, E. Teller, “Equations of state calculations by fast computing machines,” *Journal of Chemical Physics*, vol. 21, no. 6, pp. 1087–1092, Jun. 1953.