21st International Congress on Modelling and Simulation, Gold Coast, Australia, 29 Nov to 4 Dec 2015
www.mssanz.org.au/modsim2015

A Reusable Scientific workflow for conservation Planning

S. M. Guru[a], R. G. Dwyer[b], M. E. Watts[c], M. N. Dinh[d], D. Abramson[d], H. A. Nguyen[d], H. A. Campbell[e], C. E. Franklin[b], T. Clancy[a] and H. P. Possingham[c]

[a]Terrestrial Ecosystem Research Network, The University of Queensland, St Lucia, QLD 4072, [b]ECO-lab, School of Biological Sciences, The University of Queensland, St Lucia, QLD 4072, [c]Centre for Biodiversity and Conservation Science, The University of Queensland, St Lucia, QLD 4072, [d]Research Computing Centre, The University of Queensland, St Lucia, QLD 4072, [e]School of Environmental and Rural Science, University of New England, Armidale, UNSW 2351.
Email: s.guru@uq.edu.au

Abstract: In order to perform complex scientific data analysis, multiple software and skillsets are generally required. These analyses can involve collaborations between scientific and technical communities, with expertise in problem formulation and the use of tools and programming languages. While such collaborations are useful for solving a given problem, transferability and productivity of the approach is low and requires considerable assistance from the original tool developers.

Any complex scientific data analysis involves accessing and refining large volumes of data, running simulations and algorithms, and visualising results. These steps can incorporate a variety of tools and programming languages, and can be constructed as a series of activities to achieve a desired outcome. This is where scientific workflows are very useful. Scientific workflows abstract complex analyses into a series of inter-dependent computational steps that lead to a solution for a scientific problem. Once constructed, the workflow can be executed repeatedly and the results reproduced with minimal assistance from the original tool developers. This improves transferability, repeatability and productivity, and reduces costs by reusing workflow components for similar problems but using different datasets. Kepler is a popular open-source scientific workflow tool for designing, executing, archiving and sharing workflows. It has the ability to couple disparate execution environments on a single platform. For example, users can run analysis steps written in Python, R and Matlab on a single platform as part of a single analysis and synthesis experiment. Kepler provides a wide variety of reusable components that perform various tasks, including data access from databases, remote system, file system and web services, and data servers, and executes these processes in a local or distributed environment. Together these functionalities provide greater flexibility for researchers to undertake complex scientific analyses compared with traditional homogeneous environments.

In this paper, we will describe a new scientific workflow based on Kepler that automates data analysis tasks for Marxan, a widely used conservation planning software. Marxan is used by over 4,200 active users in more than 180 countries to identify gaps in biodiversity protection, identify cost effective areas for conservation investment and inform multiple-use zoning. Its use is expanding rapidly and this new functionality will improve the application of Marxan to various conservation planning problems. A Kepler workbench has been extended to provide functionality to invoke Marxan and execute it within a distributed environment using Nimrod/K. Our aim was to develop a reproducible, reusable workflow to generate conservation planning scenarios on the Kepler platform. The workflow components include data acquisition and pre-processing, construction of planning scenarios, generation of efficient solutions to the complex problem formulations and visualization of outputs. The workflow components are shared for reuse and re-configured to design and simulate other conservation planning applications. We also present a use case to demonstrate a Kepler Marxan workflow to design and implement conservation planning computational simulation experiments.

Keywords: *Conservation planning, scientific workflow, Marxan, optimisation*

1. INTRODUCTION

Systematic conservation planning is a process designed to assist natural resource managers in making good decisions on where and when to invest conservation resources. This includes the systematic identification of new protected areas as shown in Margules and Pressey (2000) as well as prioritising, implementing, and managing actions for the conservation of biological diversity and other natural resources described by Wilson et al. (2009). Early conservation planning relied on expert opinion to identify conservation goals, often doing so with a scoring system described in Margules and Usher (1981) and Pressey et al. (1993). However in the past 25 years, repeatable and transparent processes have been developed to guide the choice of these investments in a cost-effective manner to best ameliorate the threats to species, and the ecosystems, which they inhabit.

These processes are carried out using conservation planning software, with algorithms designed to solve the 'minimum-set problem' as described in Kirkpatrick et al. (1983). Here, the objective is to maximise species and site connectivity, whilst minimising the cost subject to achieving defined targets. Counts of a single species (or multiple groups of species) are often used as a proxy for total species richness, and costs include land acquisition, resource restoration and management expenses. Additional value is added to parcels of land that are connected rather than isolated, as site fragmentation can lead to increased management costs, edge effects and reduced connectivity. Scenarios based on the ecological benefits produced per dollar spent can achieve considerably greater conservation benefits than conservation plans that ignore cost, and there is broad agreement regarding the advantages of cost-benefit based return on investment analysis for conservation planning.

Marxan is the most widely-used conservation planning software, guiding resource managers and decision scientists in more than 180 countries to identify gaps in biodiversity protection, highlight cost effective areas for conservation investment and inform multiple-use zoning as described in Watts et al. (2009). Marxan operates by formulating a conservation planning scenario into an optimisation problem, and applies heuristic-based simulated annealing to find the global optimal solution. The overall objective is to minimise a combination of the cost of the reserve network and the boundary length of the entire system, whilst meeting a set of biodiversity targets as shown in Ball et al. (2009).

2. CURRENT CHALLENGES

The key steps in a Marxan analysis include the construction of essential input files (raster-based and text-based), the creation of spatial planning units (vector-based), testing of key parameters, running the optimisation solver and visualising the results. The data preparation steps are generally automated by computer programs, or implemented manually by the user using a suite of software packages. Once input data are prepared, the next challenge is to create all the input files required to execute Marxan. These input files are generated from spatial information created by the combination of conservation features, i.e. species utilisation, study area, vegetation type, and existing protected areas spatial information and land cost. Finally, Marxan runs with the input-files to provide near-optimal solutions.

The entire analysis, including input data processing, the creation of input files and Marxan simulation and visualisation, currently takes place in an asynchronous fashion. The tools used in pre-processing depend on the data type and structure, as well as the expertise of the user. These pre-processing steps are completely disjoint to Marxan and require scrutiny during the testing and sensitivity analysis phase. Since Marxan uses a heuristic-based optimisation algorithm, several solutions are generated to ensure the high dimensional phase space is examined adequately so solutions will be near optimal. As a consequence, it is common practice for conservation planning practitioners to spend considerable amount of time analysing the results.

There are several disadvantages in this approach, with multiple programming routines requiring development to create input files required for Marxan. These scripts or programs need to be managed, version controlled, documented to make them relevant, easy to use, transferable and portable. From a user's perspective this poses several challenges, including difficulty in understanding code if standard programming practices and conventions are not used, and a need to run pre-processing steps on the data before Marxan is executed. The experiments created are not easy to repeat and reproduce results without significant assistance from original tool developers. This lack of cohesive automation is an obstruction to researchers and practitioners aiming to critically assess, evaluate and interpret conservation planning experiments.

Conservation planning experiments involving Marxan are also currently constrained by the data available on conservation features. Marxan is a stand-alone command line interface application written in the C programming language. The program can be executed by providing experiment input files and algorithm parameter files. Marxan provides an output of the best solution file which lists the reserve network with the lowest objective function score provided by the simulated annealing algorithm and, the summed solution file records the selection frequency for sites across all the reserve networks generated as described in Ball et al.

(2009). Executing simulated annealing for a large number of iterations multiple times are computationally intensive. The complexity increases exponentially as the number of planning units increase as it is a high phase space problem. The simulating annealing algorithm in Marxan typically runs for 1 million iterations, with each computational experiment repeating 100 times. A high number of iterations ensure that solutions do not get stuck in local minima, and instead approaches the global minima as close as possible.

3. SCIENTIFIC WORKFLOWS

To address the need for repeatable, powerful and transferable solutions for conservation planning, we introduce a scientific workflow for Marxan that automates many of the key processing steps involved in undertaking these types of experiments. A scientific workflow is a systematic organisation of well-defined tasks/activities to achieve a scientific outcome (Talia 2013). These tasks may include retrieving data from an online data repository, feeding these data into specific algorithms for analysis, and visualising and storing results for further evaluation and assessment. Scientific workflow provides a simple way of providing high-level abstract view of a scientific experiment, while hiding the underlying detail. They are able to integrate existing software routine and scripts, services and different datasets into complex compositions that implement scientific computational experiments. Scientific workflows are also useful tools for their characteristics of repeatability and high-level graphical representation of a simulation experiments.

3.1. Kepler Workflow System

Kepler is a popular open-source scientific workflow management system for designing, executing, archiving and sharing workflows. A workflow in Kepler is a composition of different 'actors' connected through channels. Actors are the building blocks of a Kepler workflow, abstracting complex analysis into series of interdependent processing steps that leads to a final outcome. Kepler has a large pool of reusable actors, which perform specific tasks across disparate execution environments, including the invoking of web services, and tools written in the R, Python and Matlab programming language (Ludäscher et al. (2006)). Initially, Kepler was built to cater of ecology domain hence; there are large number of components (actors) that can be re-used in biodiversity, ecology and environmental science. Lists of projects using Kepler to solve various computational challenges are available in https://kepler-project.org/users/projects-using-kepler.
One of the advantages of Kepler is its ability to use distributed computing infrastructure to execute workflow tasks. Kepler has been augmented with the Nimrod/K suite to provide parallelism in the workflows as discussed in Abramson et al. (2008). Nimrod is capable of automatically spawning virtual machines for distributed jobs and terminating those machines upon completion as shown in Bethwaite et al. (2010). With Nimrod/K, users are able to develop complex workflows in which computational models are executed across changing input parameters. Nimrod/K can execute tasks with different parameters simultaneously, in dedicated cluster or on-demand cloud resources. Furthermore, Nimrod/K automates tasks such as pre-staging and transferring files into, and out of, computing resources. All the parallelisation of tasks and distribution of tasks to compute resources are hidden from a user so that the user only needs to focus on the business logic of the workflow.

4. DESIGN OF KEPLER WORKFLOW FOR MARXAN

Our aim was to develop robust and reusable Kepler scientific workflow for Marxan, which can be applied to a range of conservation planning scenarios. Our intention is that this new Kepler tool will increase the accessibility of Marxan to non-programming specialists and conservation planning practitioners. We aimed to create a Kepler scientific workflow for Marxan, encapsulating the entire workflow as a series of interrelated tasks. Once the Marxan execution is completed, a simple web app visualises the Marxan outputs interactively as maps, tables, and figures. We have developed distributed-computing enabled Kepler workflows for executing the Marxan for conservation planning problems. This enables us to run compute-intensive tasks on one or more compute resources. Two variants of workflows are implemented for any conservation planning experiments. In the first variant, Marxan will execute in a single virtual machine, or and local machine where Kepler is running. The second variant of workflow supports complete distributed computing using Nimrod. Here, multiple repetitive tasks like simulated annealing execution are sent to the cloud-based virtual machines (compute resource) for execution. This enables parallel execution of different runs of simulated annealing algorithm of same problem, and provides the flexibility to use cloud-computing resources to instantiate compute resources when required and release the resource when the execution is complete.

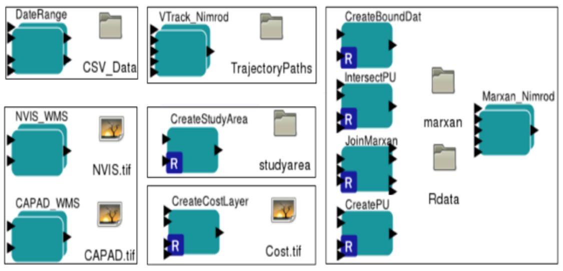

Figure 1. Different Kepler actors developed and their outputs

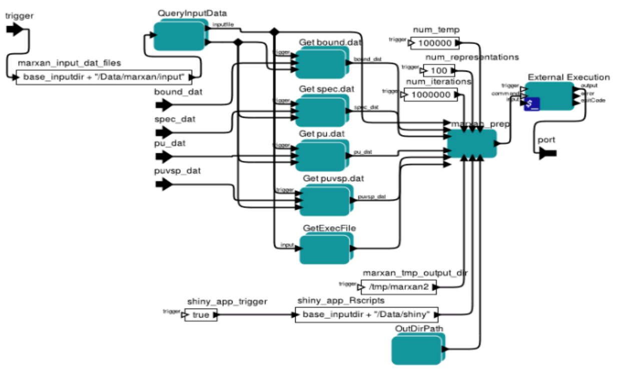

Figure 2. Kepler implementation of Marxan, this is a composite actor which is a part of Marxan implementation workflow

5. WORKFLOW IMPLEMENTATION

For the implementation of Marxan workflow, we have created Kepler actors whose tasks are to create the input files and execute Marxan. Figure 1 summarises all the actors implemented and the outputs they generate. The following functionalities are implemented as reusable actors in Kepler:
Create study area: create a raster image of the study area when a raster image and boundary of the problem space is given as input, **create cost layer**: creates a cost layer as a raster image taking input cost information and the problem space raster files. **Create planning unit**: creates a planning unit layer in an ESRI Shapefile format using study area, conservation features and planning unit granularity as input. **Web Mapping Service (WMS) access:** actors to spatially subset Web Mapping service layer to extract data overlapping the study area. Using this web service, we access National Vegetation Information System (NVIS) map products to get vegetation information about the problem space and use Collaborative Australian Protected Area Database (CAPAD) to access both marine and terrestrial protected areas in Australia. **Create boundary information**: create a Marxan boundary length file from the planning unit layer. It examines the polygon topography and creates a table illustrating all the pairs of adjacent planning units and the length of their shared boundaries. **Join pu.dat**: creates a Marxan planning unit table from the planning unit layer, cost layer, and CAPAD layer. It spatially intersects the planning unit layer with the cost and CAPAD layers to determine how much each planning unit costs, and if the planning unit is already protected. **Join spec.dat**: creates a Marxan species table from the raster maps of occurrence and NVIS layer. It lists the names of each species we are including in our Marxan scenario. It also includes key parameters (target and SPF) that require parameter testing. **Join puvsp.dat**: creates a Marxan planning unit by species matrix. The planning unit layer is intersected with the raster maps of occurrence and NVIS layers. It lists how much of each species is located in each planning unit. **Provides input.dat**: the workflow provides a template for the Marxan input parameter file. It includes the key parameter Boundary Length Modifier (BLM) that requires parameter testing. **Marxan Execution**: A Kepler actor has been developed to execute Marxan with all the required inputs. One of the advantages of Kepler is its ability to invoke programs written in different languages. We have not re-implemented Marxan but use existing executable to invoke from Kepler actor. Figure 2 shows the subset of Kepler implementation of Marxan where all the required data is created and fed into the Kepler actor, which execute Marxan.

5.1. Marxan Execution using Nimrod

As described previously, Nimrod/K enables running a Kepler actor in parallel using remote compute resources. For a task to be executed remotely, Nimrod/K automatically pre-stages input and executable files to the remote machine(s), requests processing slots and transfers output files directly to the next machine. We developed a Marxan script and use the Nimrod/K-GridJob actor to send multiple sets of input data onto multiple virtual machines on NeCTAR cloud, which are launched automatically by Nimrod. Output files are transferred back to the machine from which parallel job was instantiated. In Marxan, the simulated annealing algorithm is repeated 100 times. Instead of running on a single instance sequentially 100 times, Nimrod is used to parallelise the execution, For example, if five different machines are used to execute simulated annealing algorithm, each machine can execute 20 runs in parallel instead of one machine executing all 100 runs. This will substantially reduce the compute time. Parallelised execution of Marxan using Nimrod is shown in Figure 3.

5.2. Result Visualisation

One the Marxan execution is complete, a web app is instantiated to graphically visualise, interrogate and evaluate the results. This functionality improves the usability of Marxan and its ability to provide useful feedback which can be interpreted by a wide range of people involved in conservation planning activities, ultimately improving the quality of the decision support products.

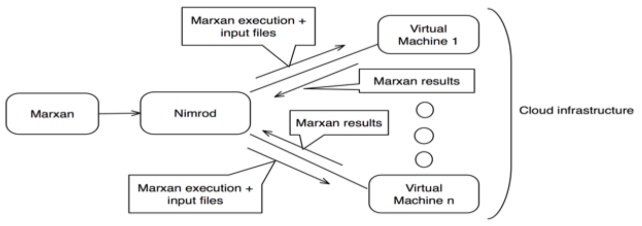

Figure 2. shows the parallelisation of Marxan execution using Nimrod

6. CASE STUDY

The rivers and estuaries of Northern Australia are highly productive environments, containing an exceptional diversity and abundance of large aquatic species. Many of the predatory species inhabiting these rivers are protected under federal, territory and state regulations. A species at particular risk is the speartooth shark *Glyphis glyphis*, which is found in tidal rivers at three distinct geographical locations in the Northern Territory and Queensland. Threats to this species include overfishing as a result of bycatch from commercial and recreational fishers, as well as habitat disturbance from mining, gas extraction and agriculture. The rarity of this species and increasing anthropogenic threats highlights a need for spatial planning solutions to mitigate the potential risks this and other threatened species living in these relatively pristine river environments.

We have implemented Kepler workflow using Marxan to generate fisheries closure scenarios that would minimise the risk of shark bycatch, whilst accommodating the needs of the local fishers. As part of the Kepler scientific workflow, records of shark presence were obtained from the IMOS/AATAMS online data repository. Data accessing and analysis actors were created to transform raw data into R and species presence data into river connectivity and river utilisation files using the V-Track R package described in Campbell et al. (2012) and available in https://cran.r-project.org/web/packages/VTrack/index.html, following Dwyer et al. (2015). These were exported as polylines and raster objects to inform construction of the Marxan input files. A cost layer representing fishing effort was constructed based on the proximity of shark records to urban areas and boat ramps. The area was also searched for existing marine and terrestrial protected areas using CAPAD (2014). These input files were then passed to a Marxan analysis and the results were visualised in an interactive Shiny application. To make the analysis run faster, certain components of the workflow were parallelised using Nimrod. These included the processing of the species presence data into planning unit and species files, as well as the execution of the simulated annealing algorithm.

Initial execution of the workflow suggests that the computation is executing successfully, from the upload of data, to the generation of optimised decisions. The ease of use and speed of analysis allowed sensitivity analysis to be conducted on the fly, and this preliminary analysis has provided the foundation upon which large-scale analysis will be undertaken to investigate optimal fisheries closures for this threatened species. Figure 4 shows the complete prototype Kepler workflow to minimise the risk of *G.glyphis* bycatch using Marxan.

7. DISCUSSION

A shareable workflow is a stepping-stone towards re-usable science where the entire computational experiment is sharable and executable by other researchers and practitioners. However, due to changes in the system hardware configuration, operating system and version of software, there is no guarantee that, once built, the Kepler scientific workflow can be executed across different platforms running various operating systems. We have used Collaborative Environment for Ecosystem Science Research and Analysis (CoESRA) to build, execute and share Marxan workflows. CoESRA provides a web-based virtual desktop environment that supports analysis and synthesis using Kepler scientific workflow as described in Guru et al. (2015)

The virtual desktop environment is accessible from a web browser (Google Chrome) lowering any barriers for accessing and using the system. Users from Australian Access Federation (AAF) affiliated institutions

can access the virtual desktop to create, build and execute Kepler workflows. Any workflows can be shared with other users by placing the workflows in a sharable folder or asking the system administrator to make the workflow sharable as part of the system.
At the time of submission, the workflow implementation for the use case described is under preliminary analysis. The workflow has significantly improved the reusability of these same processing functionalities for larger datasets in the future reducing the time taken to conduct the simulation. One of the major advantages is the parallel execution of similar tasks without changing any programming code; this has reduced the overall execution time for the experiment. A flexibility of using cloud infrastructure (both compute and storage) has provided us an access to large compute and storage resource pool enabling us to scale the conservation planning experiment to wider number of species or include more individuals from the same species. The workflow will also help us to re-run the analysis, with additional environment drivers and conservation features once they become available. One of the major advantages is the ability to access the data services; we have accessed CAPAD and NVIS datasets hosted on a TERN data server in our analysis. This has eliminated the need to host and manage third-party datasets used in the experiments.

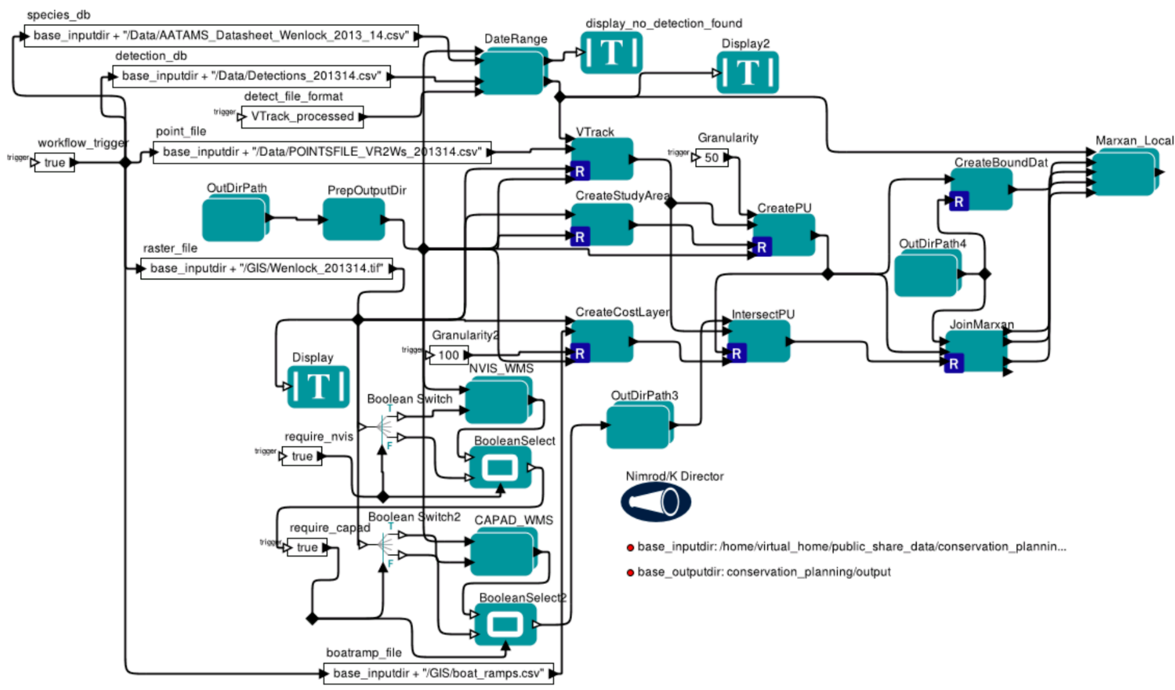

Figure 3. A complete workflow to execute conservation planning of G. glyphis

8. CONCLUSION AND FUTURE WORK

In this paper we discuss the design and development of a Kepler scientific workflow to use Marxan for conservation planning scenarios. We argue that this will enable scientists to create re-usable, sharable and repeatable experiments for conservation planning. The use of Nimrod has allowed us to parallelise the execution of simulating annealing algorithm, an optimisation algorithm used in Marxan, thus improving the overall speed of execution.
To showcase this new scientific workflow, we introduce a case study that investigates fisheries closure scenarios for a threatened shark species. While this experiment was initially run with a small subset of the species presence data, our aim is to use this as a basis for further experiments with larger and longer-term datasets. All the workflows developed are available from the CoESRA platform.
The workflow components we developed incorporate all the functionalities required to execute a Marxan analysis. However, support for more file formats and types could be incorporated in future developments to make it easier to set up experiments and generate input files for Marxan. Furthermore, additional optimisation algorithms in Marxan could be implemented to provide users with alternative techniques to solve the minimum set problem. Only by running more use cases will be able to construct very general actors, so that future scientists can configure all the required actors to set up robust experiments.
Currently, Nimrod is accessing a pool of cloud resources where the virtual machines are instantiated when any tasks need execution (on-demand). This has a lag-time due to the time taken to spawn a virtual machine

in QRIScloud infrastructure. We want to create a dedicated cluster to run Nimrod experiment so that the resources are always available to execute any tasks submitted by Nimrod.

ACKNOWLEDGEMENTS

Terrestrial Ecosystem Research Network (TERN) and Australian National Data Service (ANDS) through the Australian Government National Collaborative Research Infrastructure Strategy Program supported this project. The QRIScloud, a Queensland Node of NeCTAR and RDSI administered by Queensland Cyber Infrastructure Facility (QCIF) provides the compute and storage infrastructure, M. E. Watts and H. P. Possingham acknowledges funding received from the Australian Research Council Centre of Excellence for Environmental Decisions. The shark data used in this paper were generated as part of an ARC-Linkage Grant to C. E. Franklin and H. A. Campbell, with Australia Zoo and CSIRO as industry partners.

REFERENCE

Abramson, D., C. Enticott and I. Altinas (2008). Nimrod/K: Towards massively parallel dynamic grid workflows. In High Performance Computing, Networking, Storage and Analysis, 2008. SC 2008.

Ball, I. R., H. P. Possingham and M. Watts. (2009). Marxan and relatives: Software for spatial conservation prioritisation. In *Spatial conservation prioritisation: Quantitative methods and computational tools,* Oxford University Press.

Bethwaite, B., D. Abramson, F. Bohnert, S. Garic, C. Enticott and T. Peachey. (2010). Mixing grids and clouds: High-throughput science using the Nimrod tool family. In *Cloud computing,* 219-237.

Campbell, H. A., M. E. Watts, R. G. Dwyer and C. E. Franklin (2012). V-track: Software for analysing and visualising animal movement from acoustic telemetry detections, *Marine and Freshwater Research,* 63 815-820.

CAPAD (2014). Collaborative Australian Protected Areas Database (CAPAD) 2014, Commonwealth of Australia 2014.

Dwyer, R. G., H. A. Campbell, T. R. Irwin and C. E. Franklin (2015). Does the telemetry technology matter? Comparing estimates of aquatic animal space-use generated from gps-based and passive acoustic tracking, *Marine and Freshwater Research,* 66 654-664.

Guru, S. M., H. A. Nguyen, S. Banihit, M. Mulholland, K. Olsson and T. Clancy (2015). Development of cloud-based virtual desktop environment for synthesis and analysis for ecosystem science community. accepted in International Workshop on Science Gateways, eResearch Australasia 2015.

Kirkpatrick, S., C. D. Gelatt and M. P. Vecchi (1983). Optimization by simulated annealing, *Science,* 220 (4598), 671-680.

Ludäscher, B., I. Altintas, C. Berkley, D. Higgins, E. Jaeger, M. Jones, E. A. Lee, J. Tao and Y. Zhao (2006). Scientific workflow management and the Kepler system, *Concurrency and Computation: Practice and Experience,* 18 (10), 1039-1065.

Margules, C. R. and R. L. Pressey (2000). Systematic conservation planning, *Nature,* 405 (6783), 243-253.

Margules, C. and M. B. Usher (1981). Criteria used in assessing wildlife conservation potential: A review, *Biological Conservation,* 21 (2), 79-109.

Pressey, R. L., C. J. Humphries, C. R. Margules, R. I. Vane-Wright and P. H. Williams (1993). Beyond opportunism: Key principles for systematic reserve selection, *Trends in Ecology & Evolution,* 8 (4), 124-128.

Talia, D. (2013). Workflow systems for science: Concepts and tools, *ISRN Software Engineering,* 2013 15.

Watts, M. E., I. R. Ball, R. S. Stewart, C. J. Klein, K. Wilson, C. Steinback, R. Lourival, L. Kircher and H. P. Possingham (2009). Marxan with zones: Software for optimal conservation based land- and sea-use zoning, *Environmental Modelling & Software,* 24 (12), 1513-1521.

Wilson, K. A., J. Carwardine and H. P. Possingham (2009). Setting conservation priorities, *Annals of the New York Academy of Sciences,* 1162 (1), 237-264.