PROBLEM STATEMENT
Chips are becoming increasingly parallel, meaning that more scheduling decisions have to be made. We must now manage both spatial and temporal resource allocation of shared physical resources. At the same time, the growing prevalence of mobile devices has made power and energy first class citizens in system management. How can we get efficient execution on a diversity of platforms with applications that have different resource usage patterns?

The combinatorial scheduling problem worsens if all possible allocations of resources have to be tested at runtime. Instead, we propose to predict the effects changing an allocation will have on performance, and to use these predictions to make allocation decisions. This plan requires us to create models that capture the relationship between allocations and performance.

DESIGN OVERVIEW
We collect application behaviour and use it to create predictive models. We use the models as input to the scheduler to make decisions. Model accuracy is enhanced by the performance isolation provided by a set of hardware partitioning mechanisms.

PARTITIONING MECHANISMS
Core Partitioning: Easily partitioned by assigning threads to cores in a partition. Application chooses which threads run on which cores.

Cache Capacity Partitioning (for shared caches): Caches can be partitioned by ways or banks. For manycore chips we can use bank based, allowing an application can be allocated more local banks.

Bandwidth Partitioning: Using Globally Synchronous Frames (Lee et al. ISCA 2008) we can guarantee minimum bandwidth (Packets/Frame) and bound maximum delay, while also providing differentiated services.

MODEL FORMULATION
We create models from samples of performance data, and use them to predict the performance of allocations not included in the original sample. The inputs to our models are performance and activity metrics. The outputs of our models are predictions of metric values for untested allocations.

Linear additive model:
\[ y(x) = a_0 + \sum_{i=1}^{N} a_i x_i \]

Multivariate response surface model:
\[ y(x) = a_0 + \sum_{i=0}^{N} a_i x_i + \sum_{i=0}^{N} a_{ij} x_i x_j + \sum_{i=0}^{N} a_{i^2} x_i^2 + \ldots \]

MAKING SCHEDULING DECISIONS
We define an objective function that uses the predictive models of the two applications Experiment with different objective functions to represent best system performance, and lowest energy. We can give weights to the model outputs and other features. We use the active-set algorithm for nonlinear constrained optimization (fmincon in Matlab).

METHODOLOGY
We use Virtutech Simics with custom modules supporting hardware partitioning to collect performance data to create the models. We created synthetic benchmarks with varying types of resource requirements to explore the space of possible behaviours.

SYNTHETIC BENCHMARKS

FUTURE WORK
- Retrain models on different sample sizes
- Try making decisions with heuristic search
- Improve realism of energy model
- Make temporal as well as spatial decisions

MODEL ACCURACY
Table shows the mean accuracy (standard dev. accuracy) of the response surface model. Some metrics of performance were much easier to predict than others. Outliers severely degrade mean accuracy.

DECISION-MAKING RESULTS
Decisions made when allocating resources between two benchmarks. In red cases, the decisions made based on the models are counter to what we expected. In orange cases it was unclear whether the decisions made were correct or not. Model inaccuracy results in poor decisions.