In 2013, U.S. data centres accounted for 2.2% of the country's total electricity consumption, a figure that is projected to increase rapidly over the next decade. A significant proportion of power consumed within a data centre is attributed to the servers, and a large percentage of that is wasted as workloads compete for shared resources. Many data centres host interactive workloads (e.g., web search or e-commerce), for which it is critical to meet user expectations and user experience, called Quality of Service (QoS). There is also a wish to run both interactive and batch workloads on the same infrastructure to increase cluster utilisation and reduce operational costs and total energy consumption. Although much work has focused on the impacts of shared resource contention, it still remains a major problem to maintain QoS for both interactive and batch workloads. The goal of this thesis is twofold. First, to investigate how, and to what extent, resource contention has an effect on throughput and power of batch workloads via modelling. Second, we introduce a scheduling approach to determine on-the-fly the best configuration to satisfy the QoS for latency-critical jobs on any architecture.

To achieve the above goals, we first propose a modelling technique to estimate server performance and power at runtime called *Runtime Estimation of Performance and Power* (**REPP**). REPP's goal is to allow administrators' control on power and performance of processors. REPP achieves this goal by estimating performance and power at multiple hardware settings (dynamic frequency and voltage states (DVFS), core consolidation and idle states) and dynamically sets these settings based on user-defined constraints. The hardware counters required to build the models are available across architectures, making it architecture agnostic.

We also argue that traditional modelling and scheduling strategies are ineffective for interactive workloads. To manage such workloads, we propose **Hipster** that combines both a heuristic, and a reinforcement learning algorithm to manage interactive workloads. Hipster's goal is to improve resource efficiency while respecting the QoS of interactive workloads. Hipster achieves its goal by exploring the multicore system and DVFS. To improve utilisation and make the best usage of the available resources, Hipster can dynamically assign remaining cores to batch workloads without violating the QoS constraints for the interactive workloads.

We implemented REPP and Hipster in real-life platforms, namely 64-bit commercial (Intel SandyBridge and AMD Phenom II X4 B97) and experimental hardware (ARM big.LITTLE Juno R1). After obtaining extensive experimental results, we have shown that REPP successfully estimates power and performance of several single-threaded and multiprogrammed workloads. The average errors on Intel, AMD and ARM architectures are, respectively, 7.1%, 9.0%, 7.1% when predicting performance, and 8.1%, 6.5%, 6.0% when predicting power. Similarly, we show that when compared to prior work, Hipster improves the QoS guarantee for Web-Search from 80% to 96%, and for Memcached from 92% to 99%, while reducing the energy consumption by up to 18% on the ARM architecture.