Note: This review is likely longer than required for ECE299. The review should not exceed one page per class. Since we read approximately two papers per class, on average, a half-page review per paper is sufficient.

Title: [[Hidden for anonymity]]

Summary. The paper presents an adaptive processor front-end and an adaptive cache to navigate performance and power trade-offs. These microarchitectural trade-offs are analyzed with respect to dynamic voltage and/or frequency scaling (DVFS or DFS). As voltage scaling slows, DVFS may become less effective and such microarchitectural techniques will be increasingly important.

Strengths. The paper considers adaptivity as a knob that complements DVFS. While the authors leverage previously proposed techniques to implement an adaptive front-end and an adaptive cache, the aggregation of these results and the comparison against more pervasive DVFS and DFS schemes are useful.

Weaknesses. A discussion of control mechanisms is missing.

Detailed Comments. DVFS encounters diminishing marginal returns in power efficiency as threshold voltage hits a lower bound. Cubic power savings from voltage and frequency scaling (DVFS) are less likely and more modest linear power savings from frequency scaling alone (DFS) are more likely. To complement DVFS and DFS, the authors present scaling techniques for specific processor components.

The paper dynamically scales performance and power by dynamically disabling parts of the processor front-end. The paper proposes adapting the front-end to modulate how much instruction-level parallelism is supported. For example, speculation control dynamically constrains the number of inflight, unresolved branches.

Perhaps the most interesting insights arise when the adaptive front-end is compared against DFS and DVFS. An adaptive front-end is better than DFS linear power reductions and, in some cases, competitive to DVFS cubic power reductions (e.g., Figure 7). This approach to analyzing adaptivity will be increasingly important as DVFS becomes more difficult to implement in advanced process technologies.

To implement an adaptive front-end, this paper applies some previously proposed mechanisms (e.g., adaptive queue sizes, drowsy caches). The contribution of this paper is less about specific adaptive mechanisms and more about applying them in a coordinated fashion to the processor front-end.

No microarchitectural technique should be applied if it cannot beat DVFS. Comparing the adaptive front-end to DVFS trends is important. This comparison is something which prior work in adaptive architectures (surveyed by Albonesi et al. [3]) did not do. But the effectiveness of DVFS itself is evolving. Navigating these trends will be important.

The discussion of related work is not very explicit. But this paper in the processor front-end seems to complement prior work in adaptively managing arrays of execution units (e.g., WiDGET) and seems to provide a less complex alternative to more adaptive superscalar front-ends (e.g., Core Fusion). Given prior literature in this space, the authors provide useful data points and furthers the community's understanding of performance and power trade-offs in adaptive microarchitectures.

How does each of the steps in power gliding a component increase the complexity of the control? Do you envision software-generated signals that increase the level of power gliding for each component that

supports power gliding? Alternatively, is it more feasible to limit the number of power gliding steps that an individual component can contribute, and instead choose a handful of power settings that enable the steps across components that have provided the best improvement in the power-performance curve?