

UNITED STATES PATENT AND TRADEMARK OFFICE

---

BEFORE THE PATENT TRIAL AND APPEAL BOARD

---

MICROSOFT CORPORATION,  
Petitioner,

v.

THROUGHPUTER, INC.,  
Patent Owner.

---

IPR2022-00527  
Patent 9,632,833 B2

---

Before MICHAEL R. ZECHER, JOHN A. HUDALLA, and  
MATTHEW S. MEYERS, *Administrative Patent Judges*.

MEYERS, *Administrative Patent Judge*.

DECISION  
Granting Institution of *Inter Partes* Review  
35 U.S.C. § 314

## I. INTRODUCTION

### A. *Background and Summary*

Microsoft Corporation (“Petitioner”) filed a Petition requesting an *inter partes* review of claims 10, 13–15, 17–22, 24, 25, 27, 29, 32–34, and 37 (the “challenged claims”) of U.S. Patent 9,632,833 B2 (Ex. 1001, the “’833 patent”) pursuant to 35 U.S.C. § 311 *et seq.* Paper 2 (“Petition” or “Pet.”). ThroughPuter, Inc. (“Patent Owner”) filed a Preliminary Response. Paper 7 (“Prelim. Resp.”).

We have authority to determine whether to institute an *inter partes* review. *See* 35 U.S.C. § 314; 37 C.F.R. § 42.4(a). Under 35 U.S.C. § 314(a), we may not authorize an *inter partes* review unless the information in the petition “shows that there is a reasonable likelihood that the petitioner would prevail with respect to at least 1 of the claims challenged in the petition.” For the reasons that follow, we institute an *inter partes* review as to claims 10, 13–15, 17–22, 24, 25, 27, 29, 32–34, and 37 of the ’833 patent on all grounds of unpatentability presented.

### B. *Real Parties-in-Interest*

Petitioner identifies itself as the real party-in-interest. Pet. 73. Patent Owner identifies itself as the real party-in-interest. Paper 11, 2.

### C. *Related Proceedings*

The parties identify that the ’833 patent is involved in *ThroughPuter, Inc. v. Microsoft Corporation*, No. 2:22-cv-00344 (W.D. Wash). Paper 10, 1; Paper 11, 2. Patent Owner indicates that, “[a]s of August 18, 2022, the related district court litigation has been stayed pending resolution of the present *inter partes* review proceeding.” Paper 11, 2.

We also note that Petitioner has challenged other patents owned by Patent Owner in IPR2022-00528, IPR2022-00574, IPR2022-00757, and IPR2022-00758.

*D. The '833 Patent*

The '833 patent is titled “Scheduling Application Instances to Processor Cores Over Consecutive Allocation Periods Based on Application Requirements.” Ex. 1001, code (54). The '833 patent pertains to “techniques for improving information processing efficiency and performance through dynamically adapting processing resource types to match processing task types.” *Id.* at 1:37–41. In this regard, the '833 patent discloses

Objectives for the core 120 allocation algorithm 310 include maximizing the processor 100 core utilization (i.e., generally minimizing, and so long as there are ready [application-task/instances], eliminating core idling), while ensuring that each application 220 gets at least up to its entitled (e.g. a contract based minimum) share of the processor core capacity whenever it has processing load to utilize such amount of cores.

*Id.* at 13:1–7.

The '833 patent describes that “the range of processing jobs for a given instance of processing hardware (e.g. a server blade used for computing Infrastructure-as-a-Service), especially over the lifetime of the given hardware instance, may comprise several types of jobs and their tasks, with each type best suited for its corresponding, distinct, processor type.” Ex. 1001, 1:59–64. The '833 patent further describes an array of hardware processing units, including different processor types, will likely run several types of jobs, each best suited to different processor types. *Id.* at 1:64–2:3.

The '833 patent describes that it is quite impractical to predict “when a given instance of processing (having a hardware with a processor core or array of them) is deployed for service, to know what would be the optimal type of core for any given processor instance,” and as such, the '833 patent identifies that conventional scheduling techniques often lead to jobs “processed often by suboptimal types of processing cores.” *Id.* at 2:4–17.

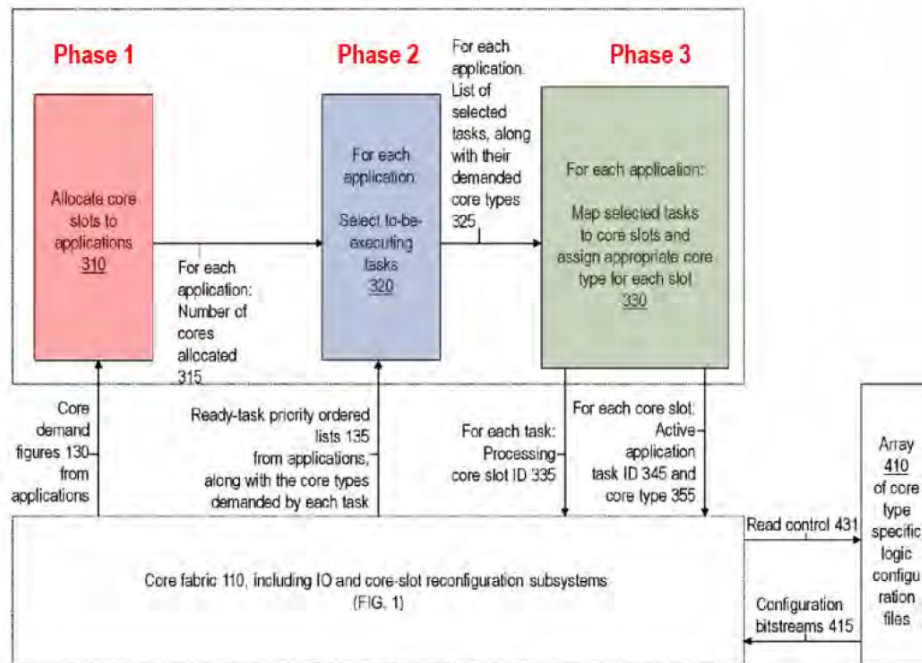
The '833 patent “provides a processing task type adaptive manycore processor system for executing time variable sets of processing software program tasks of differing types on their assigned cores of matching types.” *Id.* at 4:1–9. The '833 patent thus provides “application processing load adaptive allocation of the cores among the software applications configured for the system, as well as for dynamically reconfiguring the core slots according to the types of the processing tasks assigned to any given core slot of the multi-core fabric.” *Id.* at 8:44–53.

To accomplish this, the '833 patent utilizes for each application “capacity demand indicators 130” or “core-demand-figures (CDFs)” that “express how many cores 120 their associated app-task is presently able [to] utilize for its ready to execute instances.” Ex. 1001, 9:1–7. The system

repeatedly selects and places the to-be-executing instances 240 of the set of locally hosted app-tasks 210 to their assigned target core slots 120, as well as determines, for the core-slot reconfiguration subsystem of the processor 100 (FIGS. 3 and 4), the demanded core types for the core slots of the array 115, along with notifications of changes in the demanded core type for each given core slot.

*Id.* at 10:14–21.

Petitioner has supplied an annotated version of Figure 3, reproduced below, which is a block diagram of the system of the '833 patent.



Pet. 9 (reproducing Figure 3 of the '833 patent). Figure 3 shows algorithm 310, which allocates core slots to applications, application selection step 320, and core mapping step 330. Ex. 1001 at 13:29–14:44.

### *E. Illustrative Claim*

Petitioner challenges claims 10, 13–15, 17–22, 24, 25, 27, 29, 32–34, and 37 of the '833 patent. Pet. 2. Challenged claims 10, 15, 19, 29, and 34 are independent claims. Claims 10 and 15 are illustrative of the challenged claims, and are reproduced below, with Petitioner's bracketing, labels, and formatting.

[10.P] A method for controlling execution of a set of software programs on an array of processor cores, with a given core among the array referred to as a core slot, the method comprising:

[10.1] monitoring capacity demand indicators of one or more programs among said set of programs, with said indicator of a given one of the programs expressing a number of instances that the given program has available for execution for a succeeding Core Allocation Period (CAP);

[10.2] allocating the array of cores among the set of programs for the succeeding CAP at least in part based on said capacity demand indicators for that CAP, to produce a new allocation of the cores among the programs to replace a present allocation of the core slots among the programs; and

[10.3.i] based at least in part on the allocating, placing instances of the programs to the array of cores through sub-steps of: (i) identifying the following:

[10.3.ii] a) a set of instances of the programs from the new allocation that were not included in the present allocation, with this set referred to as activating program instances;

[10.3.iii] b) a set of instances of the programs from the present allocation that are not in the new allocation, with this set referred to as deactivating program instances; and

[10.3.iv] c) a set of core slots among the array that were assigned to the set of deactivating program instances in the present allocation, with this set referred to as available cores; and

[10.4] (ii) assigning the array of cores among the instances of the programs by placing each of the activating program instances to one of the available cores, while keeping each such program instance, which was included both in the present and the new allocation, assigned for the CAP corresponding to the new allocation to the same core as it was assigned on the CAP corresponding to the present allocation.

Ex. 1001, 21:4–42; Pet. vi–vii.

[15.P] A method for assigning instances of software programs to an array of processor cores comprising:

[15.1] for each of a series of successive core allocation periods (CAPs), selecting, from a group of executable instances of a set of software programs, a subset of the executable instances, referred to as selected instances, for execution on the cores of the array for an upcoming CAP,

[15.2] wherein the selection of the selected instances is based, at least in part, on a respective capacity demand indication of each of the set of software programs, with said indication of a given program (a) being based at least in part on a number of its executable instances that presently have input data available for processing and (b)

- indicating processor core types demanded by its executable instances;
- [15.3] assigning each of the selected instances for execution on a processor core in the array of processor cores based, at least in part, on matching the respective demanded processor core types associated with the selected instances with types of processor cores available for assignment; and
- [15.4] executing the selected instances on their assigned cores over the next CAP, at least in part, to process the input data.

Ex. 1001, 21:60–22:16; Pet. vii.

*F. Prior Art and Asserted Grounds*

Petitioner asserts that claims 10, 13–15, 17–22, 24, 25, 27, 29, 32–34, and 37 of the '833 are unpatentable based on the following grounds:

| Claims Challenged                               | 35 U.S.C. §         | References                                |
|-------------------------------------------------|---------------------|-------------------------------------------|
| 15, 17, 18, 34, 37                              | 103(a) <sup>1</sup> | Agrawal <sup>2</sup> , Brent <sup>3</sup> |
| 10, 13, 14, 19–22,<br>24, 25, 27, 29, 32,<br>33 | 103(a)              | Agrawal, Brent, Feitelson <sup>4</sup>    |

---

<sup>1</sup> The Leahy-Smith America Invents Act (“AIA”) included revisions to 35 U.S.C. §§ 102 and 103 that became effective on March 16, 2013. Because the '833 patent claims priority as a continuation to an application filed before March 16, 2013, we apply the pre-AIA versions of the statutory bases for unpatentability. See Ex. 1001, code (63).

<sup>2</sup> Kunal Agrawal et al., “Adaptive Scheduling with Parallelism Feedback,” Proceedings of the 2006 ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming, 2006 (Ex. 1004, “Agrawal”).

<sup>3</sup> US 2010/0131955 A1, published May 27, 2010 (Ex. 1005, “Brent”).

<sup>4</sup> Dror G. Feitelson, “Job Scheduling in Multiprogrammed Parallel Systems,” IBM Research Report RC 19790 (87657), 2nd Rev., August 1997 (Ex. 1006, “Feitelson”).

To support its Petition, Petitioner proffers a Declaration of Jeffrey S. Chase, Ph.D. (Ex. 1003, “Chase Decl.”).

## II. ANALYSIS

### A. *Legal Standards*

“In an [*inter partes* review], the petitioner has the burden from the onset to show with particularity why the patent it challenges is unpatentable.” *Harmonic Inc. v. Avid Tech., Inc.*, 815 F.3d 1356, 1363 (Fed. Cir. 2016) (citing 35 U.S.C. § 312(a)(3) (2012) (requiring *inter partes* review petitions to identify “with particularity . . . the evidence that supports the grounds for the challenge to each claim”)). Except in limited circumstances not present here, this burden of persuasion does not shift to the patent owner. *See Dynamic Drinkware, LLC v. Nat’l Graphics, Inc.*, 800 F.3d 1375, 1378 (Fed. Cir. 2015) (discussing the burden of proof in *inter partes* review).

A claim is unpatentable under 35 U.S.C. § 103(a) if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which that subject matter pertains. *KSR Int’l Co. v. Teleflex Inc.*, 550 U.S. 398, 406 (2007). The question of obviousness is resolved on the basis of underlying factual determinations including: (1) the scope and content of the prior art; (2) any differences between the claimed subject matter and the prior art; (3) the level of ordinary skill in the art; and (4) objective evidence of

nonobviousness when presented.<sup>5</sup> *Graham v. John Deere Co.*, 383 U.S. 1, 17–18 (1966).

In analyzing the obviousness of a combination of prior art elements, it can be important to identify a reason that would have prompted one of skill in the art “to combine . . . known elements in the fashion claimed by the patent at issue.” *KSR*, 550 U.S. at 418. A precise teaching directed to the specific subject matter of a challenged claim is not necessary to establish obviousness. *Id.* Rather, “any need or problem known in the field of endeavor at the time of invention and addressed by the patent can provide a reason for combining the elements in the manner claimed.” *Id.* at 420. Accordingly, a party who petitions the Board for a determination of unpatentability based on obviousness must show that “a skilled artisan would have been motivated to combine the teachings of the prior art references to achieve the claimed invention, and that the skilled artisan would have had a reasonable expectation of success in doing so.” *In re Magnum Oil Tools Int’l, Ltd.*, 829 F.3d 1364, 1381 (Fed. Cir. 2016) (quotations and citations omitted).

We analyze the challenges presented in the Petition in accordance with the above-stated principles.

#### *B. Level of Ordinary Skill in the Art*

We review the grounds of unpatentability presented in view of what a person of ordinary skill in the art would have understood at the time of invention. *Graham*, 383 U.S. at 17. Petitioner asserts that a person of

---

<sup>5</sup> The present record does not include any evidence of secondary considerations of nonobviousness.

ordinary skill in the art at the time of the invention “would have had a Master’s degree in computer science, computer engineering, or a related field, and 2–3 years of practical computer programming or engineering experience, including experience designing or researching parallel processing systems.” Pet. 10–11 (citing Ex. 1003 ¶¶ 18–20). Petitioner adds that “[a]dditional graduate education could substitute for professional experience, or significant experience in the field could substitute for formal education.” *Id.* at 11. Patent Owner does not dispute Petitioner’s definition of the level of ordinary skill at this time. *See generally* Prelim. Resp.

Accordingly, for purposes of this Decision, we adopt Petitioner’s description of the level of skill in the art. On the present record, we are satisfied that this definition comports with the level of skill necessary to understand and implement the teachings of the ’833 patent and the asserted prior art.

### *C. Claim Construction*

In an *inter partes* review, we construe patent claims using the same claim construction standard that would be used to construe the claim in a civil action under 35 U.S.C. § 282(b). 37 C.F.R. § 42.100(b). This rule adopts the same claim construction standard used by Article III federal courts, which follow *Phillips v. AWH Corp.*, 415 F.3d 1303 (Fed. Cir. 2005) (*en banc*), and its progeny. Under that standard, the words of a claim are generally given their “ordinary and customary meaning,” which is the meaning the term would have to a person of ordinary skill at the time of the invention, in the context of the entire patent including the specification. *See Phillips*, 415 F.3d at 1312–13. “[W]here a party believes that a specific term has meaning other than its plain meaning, the party should provide a

statement identifying a proposed construction of the particular term and where the disclosure supports that meaning.” Consolidated Trial Practice Guide,<sup>6</sup> 44.

Petitioner asserts that “[a]ll claim terms should be construed according to the *Phillips* standard. Pet. 11. In addition, Petitioner submits that the term “instance” should be construed to mean “a program ‘task,’” and that in the case of the plural “instances,” “there is no requirement in the claims for the ‘instances’ to be instances of the *same* ‘task.’” Pet. 11 (citing Ex. 1003 ¶¶ 48–50). Patent Owner does not comment on Petitioner’s proposed construction, or otherwise address claim construction. *See generally* Prelim. Resp.

We determine, at this stage, that no explicit construction of any terms is needed to resolve the issues presented by the arguments and evidence of record to reach a decision on institution. *See Realtime Data, LLC v. Iancu*, 912 F.3d 1368, 1375 (Fed. Cir. 2019) (“The Board is required to construe ‘only those terms . . . that are in controversy, and only to the extent necessary to resolve the controversy.’” (quoting *Vivid Techs., Inc. v. Am. Sci. & Eng’g, Inc.*, 200 F.3d 795, 803 (Fed. Cir. 1999))).

*D. Alleged Obviousness over Agrawal and Brent*

Petitioner asserts that claims 15, 17, 18, 34, 37 would have been unpatentable as obvious over Agrawal and Brent. Pet. 13–38. Petitioner also relies on the testimony of Dr. Chase to support its arguments. *Id.* (citing Ex. 1003). Patent Owner responds to Petitioner’s assertions. Prelim. Resp. 1–15.

---

<sup>6</sup> Available at <https://www.uspto.gov/TrialPracticeGuideConsolidated>.

*1. Overview of Agrawal (Ex. 1003)*

Agrawal is titled “Adaptive Scheduling with Parallelism Feedback.” Ex. 1004, 1.<sup>7</sup> Agrawal relates to multiprocessor computer system scheduling in a shared multiprogramming environment. *Id.* Agrawal presents “an adaptive task scheduler for multitasked jobs with dependencies that provides continual parallelism feedback to the job scheduler in the form of requests for processors.” *Id.* Agrawal focuses on “space-sharing” for parallel jobs, using a two-level strategy: “a kernel-level job scheduler which allots processors to jobs, and a user-level task scheduler which schedules the tasks belonging to a given job onto the allotted processors.” *Id.*

Using “parallelism feedback,” Agrawal’s scheduler “adaptively changes the allotment of processors according to the availability of processors in the current system environment and the job scheduler’s administrative policy.” Ex. 1004, 2. Agrawal’s adaptive task scheduler does not focus on job scheduling, but rather on task scheduling, and, “[i]nstead of using instantaneous parallelism,” Agrawal’s A-Greedy algorithm “provides parallelism feedback to the job scheduler based on a single summary statistic and the job’s behavior on the previous quantum.” *Id.* Agrawal describes that its A-Greedy algorithm “provides parallelism feedback using the past behavior of the job” and does “not assume that the job’s future parallelism is correlated with its history of parallelism.” *Id.* According to Agrawal:

Between quanta  $q - 1$  and  $q$ , the task scheduler determines its job’s *desire*  $d_q$ , which is the number of processors the job wants for quantum  $q$ . The task scheduler provides the desire  $d_q$  to the

---

<sup>7</sup> Exhibit 1004 has no original page numbers in the filed version. We rely on the page numbers inserted by Petitioner in the bottom, middle portion of each page in Exhibit 1004.

job scheduler as its parallelism feedback. The job scheduler follows some processor allocation policy to determine the **processor availability**  $p_q$ , which is the number of processors the job is entitled to get for the quantum  $q$ .

*Id.* Agrawal discloses that “[t]he number of processors the job receives for quantum  $q$  is the job’s **allotment**  $a_q = \min \{d_q, p_q\}$ , which is the smaller of its desire and the processor availability.” *Id.*

Agrawal also describes “instantaneous parallelism,” which relies on “the number of processors the job can effectively use at the current moment, as the parallelism feedback to the job scheduler.” Ex. 1004, 2. Agrawal discloses that, “[a]lthough using instantaneous parallelism for parallelism feedback is simple, it can cause gross misallocation of processor resources.”

*Id.* According to Agrawal,

the parallelism of a job may change substantially during a scheduling quantum, alternating between parallel and serial phases. The sampling of instantaneous parallelism at a scheduling event between quanta may lead the task scheduler to request either too many or too few processors depending on which phase is currently active, whereas the desirable request might be something in between. Consequently, the job may systematically waste processor cycles on the one hand or take too long to complete on the other.

*Id.* Agrawal acknowledges that A-Greedy “operates nearly as efficiently” as a “task scheduler that uses instantaneous parallelism as feedback,” and that a scheduler using instantaneous parallelism wastes no processor cycles. *Id.*

## 2. Overview of Brent (Ex. 1005)

Brent is titled “Highly Distributed Parallel Processing on Multi-Core Device.” Ex. 1005, code (54). Brent relates to multi-core processing systems, and discloses a highly distributed multi-core system with an adaptive scheduler. *Id.* ¶¶ 3, 8.

Figure 1 of Brent, reproduced below, shows an exemplary multi-core system with an adaptive scheduler.

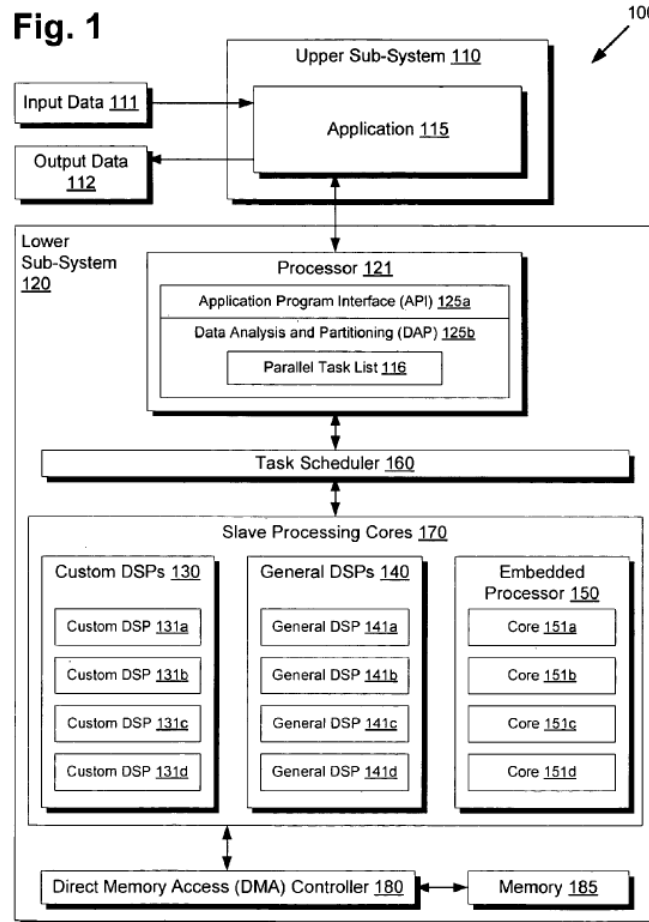


Figure 1 shows, among other things, input data 111, application 115, and processor 121 which creates task list 116, which is passed to task scheduler 160 for execution on slave processing cores 170. Brent discloses that “slave processing cores 170 may include several different types of processing cores. Custom digital signal processors (DSPs) 130 include custom DSPs 131a–131d, which may have limited instruction sets optimized for specific processing patterns.” *Id.* ¶ 16. Brent describes that “each task may be assigned a preferred core type for optimal execution.” *Id.* ¶ 18.

### 3. *Analysis of Independent Claims 15 and 34*

Petitioner asserts that claim 15 would have been obvious over Agrawal and Brent. Pet. 19–37. We use Petitioner’s notations to identify the claim elements.

- a) *[15.P] A method for assigning instances of software programs to an array of processor cores comprising:*<sup>8</sup>

Petitioner asserts that, if the preamble is treated as limiting, the combination of Agrawal and Brent discloses its contents. Pet. 24 (citing Ex. 1003 ¶¶ 71–74). More particularly, Petitioner asserts that “Agrawal discloses ‘two-level scheduling’ in which, at each scheduling quantum, a ‘job scheduler ... allots processors to jobs,’ and a ‘task scheduler [] schedules the tasks belonging to a given job onto the allotted processors.’” *Id.* at 24–25 (citing Ex. 1004, 1–2). Petitioner further asserts that Brent discloses task schedulers “that ‘distribut[e]’ a selected ‘subset of tasks’ to a[] collection of ‘slave processing cores 270.’” *Id.* at 25 (citing Ex. 1005 ¶¶ 29, 16–21, Figs. 1–3). According to Petitioner, in the asserted combination, “task schedulers for a set of jobs that were allocated cores in a scheduling quantum assign selected ‘tasks’ (*instances*) of the jobs (*software programs*) to an array of processing cores, e.g., cores 270.” *Id.* (citing Ex. 1003 ¶ 72, n.1).

---

<sup>8</sup> We need not decide whether the preamble of claim 15 is limiting for purposes of this Decision, because Petitioner has demonstrated a reasonable likelihood of showing that the cited art teaches the preamble.

- b) [15.1] *for each of a series of successive core allocation periods (CAPs), selecting, from a group of executable instances of a set of software programs, a subset of the executable instances, referred to as selected instances, for execution on the cores of the array for an upcoming CAP;*

Petitioner asserts that Agrawal's two-level scheduling operations "periodically repeat for each of a series of successive scheduling quanta (**core allocation periods (CAPs)**)." Pet. 26 (citing Ex. 1005, 2). Petitioner further asserts that during each quanta, Agrawal's "task scheduler 'schedules the tasks belonging to a given job onto the allotted processors.'" *Id.* (citing Ex. 1004, 2). Petitioner further asserts that, "[a]s Brent explains, scheduling tasks of a job, e.g., from an application 115, onto an array of processor cores involves selecting a subset of tasks for execution on the cores." *Id.* (citing Ex. 1005 ¶¶ 15, 20, 27).

According to Petitioner, the combination resulting from Agrawal and Brent "would provide a job scheduler that received parallelism feedback from each of the jobs' respective task schedulers," "allot a set of cores (processors) to each job based on the job's desired number of cores indicated in its parallelism feedback," and for each job, the task scheduler "would then assign the job's ready-to-execute task(s) to respective ones of the cores allotted to the job, and based on Brent, seek to match tasks to their preferred core types according to the tasks' type-affinity indicators." Pet. 21 (citing Ex. 1003 ¶ 65; Ex. 1004, 1–2; Ex. 1005 ¶¶ 21–23, 27–29, Figs. 1–3).

Petitioner explains, that in Agrawal,

"desire" is "the number of processors the job wants for quantum  $q$ "; "allotment" is "[t]he number of processors the job receives for quantum  $q$  ..., which the smaller of its desire and the processor availability", [and] ("**instantaneous parallelism**" is "the number of processors the job can effectively use at the current moment").

*Id.* at 28 (citing Ex. 1004, 2). Petitioner reasons

[t]o accommodate an allotment of fewer cores to the jobs than desired (e.g., fewer than the number of “ready-to-execute” tasks that the “job [could] effectively use at the current moment”), the respective task schedulers of Jobs 1 and 2 select a subset of their tasks for execution during the scheduling quantum.

*Id.* Relying on the declaration testimony of Dr. Chase, Petitioner concludes, that because

this same process is repeated for each successive scheduling quantum (**CAP**) based on the teachings of Agrawal, the Agrawal-Brent combination would select in each CAP, from a group of executable instances of a set of software programs . . . , a subset of the executable instances . . . for execution on the cores of the array for an upcoming quantum (**CAP**).

*Id.* at 28–29 (citing Ex. 1003 ¶ 78).

*c) [15.2] wherein the selection of the selected instances is based, at least in part, on a respective capacity demand indication of each of the set of software programs, with said indication of a given program (a) being based at least in part on a number of its executable instances that presently have input data available for processing and (b) indicating processor core types demanded by its executable instances;*

Petitioner asserts that, in Agrawal, “[a] task becomes **ready** to be executed when all its parents have been executed.” Pet. 29 (citing Ex. 1004, 1). Petitioner explains that Agrawal describes “instantaneous parallelism” as “feedback provided to the job scheduler from each job’s respective task scheduler [that] indicates ‘the number of processors the job can effectively use at the current moment.’” *Id.* (citing Ex. 1004, 2). According to Petitioner, one of ordinary skill in the art “would have understood, including for long-task jobs (*see* [Pet. 13–17]), that ‘the number of processors the job can effectively use at the current moment’—i.e., the measure for ‘instantaneous parallelism’—corresponds and equates to the number of tasks

that the job has ready to execute.” *Id.* at 29–30 (citing Ex. 1003 ¶¶ 79). Petitioner concludes that “[t]he parallelism feedback (*capacity demand indication*) of a given job (*given program*) in the Agrawal-Brent combination is therefore based at least in part on a number of the job’s tasks (*executable instances*) that presently have input data available for processing.” *Id.* at 30 (citing Ex. 1003 ¶¶ 54, 57). Petitioner also notes that both Agrawal and Brent describe that all dependent tasks must be resolved before a dependent task may become ready to execute. *Id.* (citing Ex. 1004, 1; Ex. 1005 ¶¶ 20, 26–27, Figs. 2–3).<sup>9</sup>

With respect to “processor core types,” Petitioner asserts that Brent discloses “an indication for an application (*a given software program*) of the processor core types demanded by its tasks (*executable instances*).” Pet. 31 (citing Ex. 1005 ¶¶ 16–18, 22, 27, 29, Fig. 2). According to Petitioner, “[i]t would have been obvious to apply Brent’s teachings for indicating demanded processor core types in combination with Agrawal.” *Id.* (citing Pet. 19–24).

More specifically, Petitioner asserts that one of ordinary skill in the art would have found it obvious to combine the teachings of Agrawal and Brent, thereby implementing a two-level scheduling strategy like that disclosed in Agrawal in the context of a system that provides an array of processing cores of various types (e.g., embedded CPUs, custom DSPs, and general DSPs) as taught in Brent.

---

<sup>9</sup> Alternatively, Petitioner asserts that this aspect of the limitation would have been obvious “in view of Brent to count a task as ready to execute in the resulting combination based on the task having input data available to process.” Pet. 31 (citing Ex. 1003 ¶ 80).

Pet. 21 (citing Ex. 1003 ¶ 65; Ex. 1005 ¶¶ 16–17). Petitioner provides several reasons why one of ordinary skill in the art would have been motivated to combine the teachings of Agrawal and Brent. Pet. 21 (citing Ex. 1003 ¶¶ 66–70). For example, Petitioner asserts that both references are directed to “scheduling tasks in a parallel processing environment, including scheduling techniques that account for task dependencies and application demands that change over time.” *Id.* at 21–22 (citing Ex. 1004, 1–3; Ex. 1005 ¶¶ 13, 20, 26). According to Petitioner, “[i]t would have been obvious to apply these concepts from Brent in combination with Agrawal to achieve predictable benefits, including an improved ability to optimize task execution on a processor whose instruction set matches the instruction types provided in a task.” *Id.* at 22 (citing Ex. 1003 ¶ 67; Ex. 1005 ¶¶ 16, 18–19).

*d) [15.3] assigning each of the selected instances for execution on a processor core in the array of processor cores based, at least in part, on matching the respective demanded processor core types associated with the selected instances with types of processor cores available for assignment; and*

Petitioner asserts that the combination of Agarwal and Brent

discloses task schedulers for each job (***software program***) that collectively assign each of the selected tasks (***selected instances***) for execution on a processor core in the array (e.g., array 170) based, at least in part, on matching the respective demanded processor core types associated with the selected tasks (***selected instances***) with types of processor cores available for assignment.

Pet. 34–35 (citing Ex. 1003 ¶¶ 85–86; Ex. 1004, 1; Ex. 1005 ¶¶ 16–23, 29; Pet. 19–24, 29–34).

- e) [15.4] *executing the selected instances on their assigned cores over the next CAP, at least in part, to process the input data*

Petitioner asserts that the combination of Agrawal and Brent “discloses executing the selected tasks (*selected instances*) on their assigned cores over the next scheduling quantum (*CAP*), at least in part, to process the input data.” Pet. 36 (citing Ex. 1003 ¶ 87; Ex. 1004, 2; Ex. 1005 ¶¶ 14, 18, 20, 26–27, Fig. 2; Pet. 19–24).

#### 4. *Patent Owner’s Contentions*

Patent Owner argues that Petitioner’s ground of unpatentability based on Agrawal and Brent is deficient. *See* Prelim. Resp. 1–10, 12–13. Patent Owner first disputes Petitioner’s proposed modification of Agrawal “to allocate cores according to the principle of ‘instantaneous parallelism.’” *Id.* at 1. According to Patent Owner, such a modification would fundamentally change Agrawal’s principle of operation, and Petitioner fails to provide adequate rationale to support the suggested change in Agrawal’s principle of operation. *Id.* at 1–7. Patent Owner next argues that Petitioner fails to establish that the “prior art discloses ‘capacity demand indicators’ reflecting a program instance’s processing needs for an upcoming core allocation.” *Id.* at 8. Patent Owner further argues that, “[e]ven if Agrawal expressed processor demand based on the data presently available for processing during the present core allocation,” Agrawal still fails to disclose that “the claimed ‘selecting’ of instances is for execution over the next CAP.” *Id.* at 13.

#### 5. *Discussion*

Having considered the positions of the parties, we conclude that, at this stage of the proceeding, Petitioner has shown a reasonable likelihood

that it would prevail in challenging the patentability of independent claim 15. In our view, Petitioner’s obviousness approach, on this record, adequately identifies where all the elements of claim 15 are found in the prior art, and, in considering claim 15 as a whole, Petitioner demonstrates adequate reasoning to combine the teachings of Agrawal and Brent.

Patent Owner first argues that the Petition is improperly “premised on the assertion that [one of ordinary skill in the art] would modify Agrawal’s principle of operation from A-GREEDY’s calculating ‘desire’ based on past processor usage to calculating ‘desire’ based on the number of processors a job could use at the current moment.” Prelim. Resp. 4. Patent Owner asserts that Agrawal’s A-GREEDY system utilizes “parallelism feedback to the job scheduler based on a single summary statistic and the job’s behavior on the *previous* quantum.” *Id.* (citing Ex. 1004, 2). Patent Owner acknowledges that Agrawal “mentions another type of feedback, referred to as ‘instantaneous parallelism,’” but argues that Agrawal’s system “specifically avoids the sampling of present processor utilization inherent in the principle of ‘instantaneous parallelism,’” and points out that Agrawal even “mentions ‘instantaneous parallelism’ as a problematic scheduling feedback strategy.” *Id.* at 3–6. Patent Owner, thus, takes the position that there is “no basis to rewrite Agrawal to use the very process it specifically sets out to avoid,” except when engaging in impermissible hindsight. *Id.* at 5.

At this stage of the proceeding, we do not agree with Patent Owner’s arguments. Patent Owner’s argument does not establish that using “instantaneous parallelism” instead of “parallelism feedback” contradicts a basic teaching in Agrawal or would otherwise change the basic principles

under which the two-level scheduling techniques disclosed in Agrawal operate. And, based on the current record, we do not see any persuasive evidence that established that providing feedback to Agrawal's job scheduler using "instantaneous parallelism," as opposed to "parallelism feedback," would render Agrawal inoperable.

Specifically, we agree with Petitioner that Agrawal "describes conventional feedback channels by which a task scheduler provides feedback to a job scheduler about the desired core allocation for a job, which Agrawal's original job scheduler already uses to inform allocation decisions." Pet. 23 (citing Ex. 1004, 1–2). We acknowledge, as does Petitioner, that "much of Agrawal's disclosure focuses on a proposed parallelism feedback solution referred to as 'A-GREEDY,'" but Agrawal also teaches that it is well-known in the art to use "'the notion of *instantaneous parallelism*' to provide feedback to the job scheduler." Pet. 15 (citing Ex. 1004, 2). In fact, Agrawal discloses that "using instantaneous parallelism for parallelism feedback is simple" (Ex. 1004, 2), and describes circumstances in which "instantaneous parallelism" feedback is more efficient to the historical-based feedback utilized in "parallelism feedback." *See id.* at 3 ("Thus, even without up-to-date information on instantaneous parallelism, A-GREEDY operates nearly as efficiently."), 7 ("If the task scheduler can communicate with the job scheduler before every vector operation, then using instantaneous parallelism as feedback works fine.").

At this stage of the proceeding, we also credit the declaration testimony of Dr. Chase, who, based on his review of Agrawal, testifies that one of ordinary skill in the art

would have recognized “instantaneous parallelism” feedback to be an especially useful form of parallelism feedback for jobs whose task durations are long relative to the length of a quantum. For short quanta (e.g.,  $L=1$ ), the parallelism of a job does not change over the course of a quantum and “adaptive scheduling can be done efficiently using instantaneous parallelism as feedback.” [Ex. 1004], 3; *see also id.*, (“[i]nstantaneous parallelism wastes none,” i.e., wastes no “processor cycles”). For longer quanta (e.g.,  $L>1$ ), instantaneous parallelism can still be an effective form of parallelism feedback if the tasks for a job are sufficiently long that they take at least a full quantum (and potentially multiple quanta) to complete execution.

Ex. 1003 ¶ 56. Accordingly, we do not understand the Petition to be rewriting Agrawal’s disclosure (Prelim. Resp. 5), but rather, we understand Agrawal’s disclosure to more broadly teach various feedback techniques related to adaptive task scheduling. *See, e.g.*, Ex. 1003 ¶ 51 (citing Ex. 1004, 1) (Agrawal “generally addresses two-level scheduling techniques for parallelizing execution of jobs on a set of processors.”). Petitioner may rely on Agrawal for all that it teaches to those of ordinary skill in the art. *In re Fritch*, 972 F.2d 1260, 1264 (Fed. Cir. 1992). Contrary to Patent Owner’s arguments, Petitioner does not appear to suggest modifying Agrawal, but instead, relies on its broader teachings as the basis for its combination with Brent. Pet. 26–34.

Patent Owner next argues that the combination of Agrawal and Brent is deficient because it fails to disclose or suggest “allocat[ing] cores based on demand expressed for a **subsequent** core allocation period.” Prelim. Resp. 2. According to Patent Owner,

the Petition does not establish that prior art discloses “capacity demand indicators” reflecting a program instance’s processing needs for an **upcoming** core allocation. Instead, the prior art—at best—mentions feedback reflecting a program instance’s **past**

and **present** processor utilization, rather than its demands for processors for the subsequent core allocation period as expressed during the present core allocation period.

*Id.* at 8. With respect to claim 15, Patent Owner contends that “the claimed ‘selecting’ of instances is for execution over the **next** CAP” (*id.* at 13), and as such, “[e]ven if Agrawal expressed processor demand based on the data **presently** available for processing during the present core allocation, that would not meet the” language of claim 15. *Id.* According to Patent Owner, this is because “Agrawal does not monitor a program instance’s processor demands for an **upcoming** core allocation period. Instead, Agrawal states that ‘instantaneous parallelism’ represents feedback from the task scheduler to the job scheduler reflecting “the number of processors the job can effectively use at the **current** moment.” *Id.* at 10.

At this stage of the proceeding, we do not agree with Patent Owner’s argument. Patent Owner asserts that Petitioner fails to identify what prior art disclosure “meets the claimed ‘capacity demand indicator.’” Prelim. Resp. 2. However, relying on the declaration testimony of Dr. Chase, Petitioner asserts that one of ordinary skill in the art “would have understood, including for long-task jobs (*see* [Pet. 13–17]), that ‘the number of processors the job can effectively use at the current moment’—i.e., the measure for ‘instantaneous parallelism’—corresponds and equates to the number of tasks that the job has ready to execute.” Pet. 29–30 (citing Ex. 1003 ¶ 79). Here, the ’833 patent discloses that “capacity demand indicators 130” also referred to as “core-demand-figures (CDFs)” “express how many cores 120 their associated app-task is presently able utilize for its ready to execute instances.” Ex. 1001, 9:4–7. Thus, we consider this a sufficient showing, at this time, that one of ordinary skill in art would have

understood Agrawal to disclose a “capacity demand indicator,” as required by independent claim 15.

To the extent Agrawal uses the term “current moment” to describe the type of feedback data provided to the job scheduler via “instantaneous feedback,” we understand this feedback data to be describing the demands for processors based on the processing period that is just about to begin, which is same as the claimed “upcoming” period. *See* Ex. 1004, 2 (“Between quanta  $q - 1$  and  $q$ , the task scheduler determines its job’s *desire*  $d_q$ , which is the number of processors the job wants for quantum  $q$ .”). Therefore, we understand the current moment in Agrawal to be the moment between  $q - 1$  and  $q$ , where  $q$  is the claimed “upcoming CAP.” Thus, for purposes of institution, we determine that Petitioner has shown sufficiently that Agarwal discloses “the claimed ‘selecting’ of instances is for execution over the *next* CAP,” as required by claim 15.

Consistent with this understanding, we disagree with Patent Owner’s argument<sup>10</sup> that Dr. Chase’s assertions related to “Agrawal operat[ing] according to the principle of ‘instantaneous parallelism’” are “conclusory and demonstrably incorrect.” Prelim. Resp. 11 (citing Pet. 44 (citing Ex. 1003 ¶ 108)). For the reasons discussed above, we find Dr. Chase’s declaration testimony to be supported by Agarwal’s disclosure (*see, e.g.*, Ex. 1004, 3 (Agrawal describing “a greedy task scheduler that uses instantaneous parallelism as feedback.”)), and, on the current record,

---

<sup>10</sup> Although Patent Owner makes this argument with respect to independent claims 10 and 29 (Prelim. Resp. 10–12), we address it here in our analysis as it is equally relevant to our analysis of independent claim 15 based on the similar subject matter.

adequate to support Petitioner and Dr. Chase’s position that “one method for computing the job’s ‘desire’ is to employ ‘instantaneous parallelism,’ which indicates ‘the number of processors the job can effectively use at the current moment[] . . . .” Pet. 44 (citing Ex. 1003 ¶ 108).

*a) Conclusion for Claim 15*

Accordingly, at this stage of the proceeding, we determine that Petitioner has established a reasonable likelihood of prevailing on its assertion that claim 15 would have been obvious over the combination of Agarwal and Brent.

*6. Independent Claim 34*

According to Patent Owner, claim 34 is similar to claim 15. *See* Prelim. Resp. 12. Patent Owner does not contest Petitioner’s challenge to claim 34 other than by relying on its arguments presented for claim 15, as discussed above. *See id.* at 12–13.

Petitioner provides an element-by-element analysis of where each element in the challenged claims is disclosed in, or would have been obvious in view of, the cited references. Pet. 38. Petitioner also relies on the testimony of Dr. Chase for evidentiary support. *Id.* (citing Ex. 1003 ¶¶ 92–99).

Accordingly, for the reasons set forth with respect to claim 15 above, and on this preliminary record, we determine that Petitioner has established a reasonable likelihood of prevailing on its assertion that claim 34 would have been obvious over the combination of Agarwal and Brent.

*7. Dependent Claims 17, 18, and 37*

Claims 17, 18, and 37 depend directly or indirectly from at least one of claims 15 and 34. Petitioner provides a clause-by-clause analysis of

where each element in dependent claims 17, 18, and 37 is disclosed in, or would have been obvious in view of, Agrawal and Brent. Pet. 37–38. Throughout Petitioner’s analysis, Petitioner cites and relies on the testimony of Dr. Chase for evidentiary support. *Id.* (citing Ex. 1003 ¶¶ 88–89, 91–99).

Patent Owner does not specifically address any of the dependent claims. *See* Prelim. Resp. 1–15.

We find Petitioner’s arguments and evidence in support thereof sufficient at this stage of the proceeding for the reasons explained by Petitioner. *See* Pet. 37–38. Accordingly, based on the arguments and evidence on the record before us, we determine, for purposes of this Decision, Petitioner has established a reasonable likelihood of prevailing on dependent claims 17, 18, and 37.

*E. Alleged Obviousness over Agrawal, Brent, and Feitelson*

Petitioner asserts that claims 10, 13, 14, 19–22, 24, 25, 27, 29, 32, and 33 would have been unpatentable as obvious over Agrawal, Brent, and Feitelson. Pet. 38–73. Petitioner also relies on the testimony of Dr. Chase to support its arguments. *Id.* (citing Ex. 1003). Patent Owner responds to Petitioner’s assertions. Prelim. Resp. 1–15.

*1. Overview of Feitelson (Ex. 1006)*

Feitelson is titled “Job Scheduling in Multiprogrammed Parallel Systems.” Ex. 1006, i. Feitelson is an IBM Research Report and concerns scheduling on parallel computer systems shared by a number of users. *Id.*, Abstr. Feitelson “describe[s] the many different approaches within a unified framework based on the mechanisms used to achieve multiprogramming,” such as single-level and two-level scheduling, time slicing, space slicing, and hybrid approaches involving both. *Id.* In Feitelson, which is primarily

concerned with “interactive use,” “the full range of schemes developed for multiprogramming on parallel machines are reviewed. This includes both paper designs from academia and practical ones that are commercially available.” *Id.* at 3. Feitelson does not discuss scheduling involving interdependencies between processors, scheduling with real-time constraints, or scheduling on “heterogeneous networks” of different computer hardware types. *Id.* at 3–4.

## 2. *Analysis of Independent Claim 10*

Petitioner asserts that claim 10 would have been obvious over Agrawal, Brent, and Feitelson. Pet. 41–58. We use Petitioner’s notations to identify the claim elements.

- a) *[10.P] A method for controlling execution of a set of software programs on an array of processor cores, with a given core among the array referred to as a core slot, the method comprising:*

Petitioner asserts that, if the preamble is treated as limiting, the combination of Agrawal and Brent discloses its contents. Pet. 41–43 (citing Ex. 1003 ¶¶ 105–106; Ex. 1004, 1–2; Ex. 1005 ¶¶ 14, 16–17, Fig. 1; Ex. 1006, 8; Ex. 1008, 261–262). Petitioner asserts that “Agrawal discloses a method for controlling execution of a set of jobs (***software programs***) on a set of processors.” *Id.* at 41 (citing Ex. 1004, 1–2; Pet. 13–17). Petitioner acknowledges that Agrawal does not disclose using any particular type of processors or describe its processors as an “array of processor cores,” but argues that such arrays were known in the art at the time of the invention, as evidenced by Brent. *Id.* at 42 (citing Ex. 1005 ¶¶ 14, 16–17, Fig. 1; Ex. 1003 ¶ 106). According to Petitioner, it would have been obvious to implement Agrawal’s “two-level scheduling” using Brent’s “array of

processor cores” “to achieve predictable benefits associated with core arrays.” *Id.* at 43 (citing Ex. 1003 ¶ 106).

b) *[10.1] monitoring capacity demand indicators of one or more programs among said set of programs, with said indicator of a given one of the programs expressing a number of instances that the given program has available for execution for a succeeding Core Allocation Period (CAP);*

Petitioner asserts that “Agrawal discloses, and the combination provides, a ‘job scheduler’ that monitors ‘parallelism feedback’ (*capacity demand indicators*) received from the respective task schedulers of each managed job (*one or more programs among said set of programs*).” Pet. 43 (citing Ex. 1003 ¶¶ 107–110; Ex. 1004, 1–2). Petitioner explains that Agarwal’s adaptive scheduling strategy utilizes a task scheduler that provides “*parallelism feedback*” to a “job scheduler so that when a job cannot use many processors, those processors can be reallocated to jobs with ample need.” *Id.* at 43–44 (citing Ex. 1004, 2). Petitioner further explains that Agarwal’s “‘parallelism feedback’ for a job indicates the ‘job’s *desire*  $d_q$ , which is the number of processors the job wants for quantum  $q$ .” *Id.* at 44 (citing Ex. 1004, 2). According to Petitioner, in Agarwal,

one method for computing the job’s “desire” is to employ “instantaneous parallelism,” which indicates “the number of processors the job can effectively use at the current moment[.] . . .” [Ex. 1004, 2]. As jobs seek to execute all of their “ready” tasks, the number of processors that a job would desire and effectively use at a given time relates to the number of processors (cores) that the job would use to execute its “ready” tasks.

*Id.* (citing Ex. 1004, 1–2).

Relying on the declaration testimony of Dr. Chase, Petitioner asserts that “Agrawal and Brent both teach scheduling concepts that assign just one task to a given core at a time,” and as such, “Agrawal’s ‘instantaneous

parallelism' feedback provides a '**capacity demand indicator**' for a job (**program**) that expresses a number of tasks (**instances**, see *supra*, [Pet. 11–13]) that the job (**program**) has available for execution in a succeeding scheduling quantum (**core allocation period**). Pet. 44 (citing Ex. 1003 ¶ 108); *see also* Ex. 1004, 3 (“Between quanta  $q - 1$  and  $q$ , the task scheduler determines its job's desire ... for quantum  $q$ .”).

- c) [10.2] *allocating the array of cores among the set of programs for the succeeding CAP at least in part based on said capacity demand indicators for that CAP, to produce a new allocation of the cores among the programs to replace a present allocation of the core slots among the programs; and*

Petitioner asserts

Agarwal discloses, and the combination provides, a “job scheduler” that allocates processors (e.g., allocates cores in an “array of cores” based on Brent) among a set of “jobs” (**programs**) for a succeeding scheduling quantum (**succeeding CAP**) at least in part based on the “desires” expressed by the jobs' respective task schedulers through parallelism feedback (**capacity demand indicators**) for that quantum.

Pet. 46 (citing Ex. 1004, 1–2; Ex. 1003 ¶ 111; Pet. 13–17, 39–41). More particularly, Petitioner asserts that Agarwal discloses that “[b]etween quantum  $q-1$  (**present CAP**) and  $q$  (**succeeding CAP**), the job scheduler produces a new allocation of the cores among the jobs to replace a present allocation of the core slots among the jobs.” *Id.* (citing Ex. 1004, 2).

Relying on the declaration testimony of Dr. Chase, Petitioner provides an example consistent with the teachings of Agarwal and Brent. *See* Pet. 47 (citing Ex. 1003 ¶ 112). In the example, Petitioner explains that “Job 1 may be allotted additional cores in the new allocation if its ‘desire’ increased for

the succeeding quantum to accommodate additional ready-to-execute tasks.”  
*Id.* (citing Ex. 1004, 1–2).

- d) [10.3i] based at least in part on the allocating, placing instances of the programs to the array of cores through sub-steps of:*  
*(i) identifying the following:*

Petitioner asserts that “Agrawal discloses that ‘[e]ach job has its own task scheduler,’ and the task scheduler ‘schedules the tasks belonging to a given job onto the allotted processors’ for a scheduling quantum.” Pet. 48 (citing Ex. 1004, 1–2; Pet. 13–17). Petitioner explains that “[s]cheduling tasks involves placing tasks (*instances*) of the various jobs (*programs*) to the respective processors allotted to the job for a given quantum.” *Id.* (citing Ex. 1005 ¶¶ 18–23, 29). According to Petitioner, the combination of Agrawal, Brent, and Feitelson would “schedule tasks for cores (core slots) in an array of processor cores by placing tasks for execution on individual cores.” *Id.* at 48–49 (citing Ex. 1003 ¶ 113; Ex. 1005 ¶¶ 18–23, 29). Petitioner further asserts that, “[a]s with Agrawal, the placement of tasks to cores in the resulting combination is ‘based at least in part on the allocating.’” *Id.* at 49 (citing Ex. 1003 ¶ 113).

- e) [10.3ii] a) a set of instances of the programs from the new allocation that were not included in the present allocation, with this set referred to as activating program instances;*

Petitioner asserts that “[i]n placing tasks to cores, task schedulers in the Agrawal-Brent-Feitelson combination identify a set of activating tasks (*activating program instances*) of the jobs (*programs*) from the new allocation for a succeeding quantum (*succeeding CAP*) that were not included in the present allocation of a preceding quantum (*preceding CAP*).”

Pet. 49 (citing Ex. 1003 ¶¶ 114–115; Ex. 1004, 1; Ex. 1005 ¶¶ 15–23, Figs. 2–3). According to Petitioner,

it would have been obvious to identify activating tasks in the Agrawal-Brent-Feitelson combination to enable the respective task schedulers of each job to assign the activating tasks to the “available cores” for that job, i.e., those cores that have been allotted to the job and that are available based on the deactivation of tasks from a preceding quantum (CAP).

*Id.* at 49–50 (citing Ex. 1003 ¶ 114; Pet. 50–58).

*f) [10.3iii] b) a set of instances of the programs from the present allocation that are not in the new allocation, with this set referred to as deactivating program instances; and*

Petitioner asserts that similar to identifying activating tasks, as recited limitation [10.3ii], the combination of Agrawal, Brent, and Feitelson discloses that task schedulers “identify a set of deactivating tasks (*deactivating program instances*) of the jobs (*programs*) from the new allocation for a succeeding quantum (*succeeding CAP*) that were not included in the present allocation of a preceding quantum (*preceding CAP*).”

Pet. 50–51 (citing Ex. 1003 ¶ 116; Ex. 1005 ¶¶ 20–23, 27–29, Fig. 2).

According to Petitioner, one of ordinary skill in the art

would have appreciated that identifying deactivating tasks and their corresponding cores would be advantageous to distinguish which allotted cores were available for activating tasks and which allotted cores were not, especially since the resulting system would predictably keep tasks that continue execution across succeeding quanta on their same cores based on the teachings of Feitelson.

*Id.* at 51 (citing Ex. 1003 ¶ 116; Pet. 53–58).

- g) [10.3iv] c) *a set of core slots among the array that were assigned to the set of deactivating program instances in the present allocation, with this set referred to as available cores; and*

Petitioner asserts that “the teachings of Agrawal, Brent, and Feitelson provide for identification of a set of available core slots (**available cores**) in an array that were assigned to a set of deactivating tasks (**deactivating program instances**) in the present allocation for a present scheduling quantum (**CAP**).” Pet. 52 (citing Ex. 1003 ¶ 117; Ex. 1005 ¶¶ 20–22, Figs. 2–3; Pet. 13–19, 39–41, 48–52).

- h) [10.4] (ii) *assigning the array of cores among the instances of the programs by placing each of the activating program instances to one of the available cores, while keeping each such program instance, which was included both in the present and the new allocation, assigned for the CAP corresponding to the new allocation to the same core as it was assigned on the CAP corresponding to the present allocation.*

Petitioner asserts that “Agrawal discloses that ‘[e]ach job has its own task scheduler,’ and the task scheduler ‘schedules the tasks belonging to a given job onto the allotted processors’ for a scheduling quantum.” Pet. 53 (citing Ex. 1004, 2). Petitioner further asserts that, in Agarwal, “[s]cheduling tasks involves assigning tasks (**instances**) of the various jobs (**programs**) to the respective processors allotted to the job for a given quantum.” *Id.* (citing Ex. 1004, 1–2; Ex. 1005 ¶¶ 18–23, 29, Fig. 2). Petitioner adds that Brent discloses a task scheduler that similarly assigns selected tasks “for execution on different cores for a ‘present time frame,’” but also allows tasks to continue executing on their previously assigned cores. *Id.* at 53–54 (citing Ex. 1005 ¶¶ 18–23, 29; Ex. 1003 ¶ 120).

To the extent Agarwal and Brent may not disclose the elements of limitation [10.4], Petitioner contends that Feitelson teaches these elements.

Pet. 54 (citing Ex. 1003 ¶¶ 122–128). Petitioner argues that Feitelson demonstrates that “an important ‘consideration in the scheduling of threads . . . is matching threads with the most appropriate PE [processing entity] for them to run on.’” *Id.* (citing Ex. 1006, 59). In this regard, Feitelson describes an “affinity scheduling” policy that “tries to schedule threads on the same PE on which they ran most recently, under the assumption that this particular PE might still have some relevant data in its cache.” *Id.* According to Petitioner, one of ordinary skill in the art

would have found it obvious to apply Feitelson’s suggestion for keeping threads (e.g., tasks) executing in successive scheduling quanta on their same processors (e.g., cores) across the successive quanta (thereby “increas[ing] the effective scheduling quantum” for the tasks), and predictable reasons would have motivated the combination.

Pet. 55 (citing Ex. 1003 ¶ 112). Relying on the declaration testimony of Dr. Chase, Petitioner asserts that, in contrast, one of ordinary skill in the art

would have understood that switching threads or tasks from one processor to another ordinarily requires overhead and additional clock cycles to interrupt ongoing execution of the thread, store data that characterizes the thread’s current execution context, and transfer the instructions, data, and other necessary information for executing the thread to the new processor.

*Id.* at 56 (citing Ex. 1003 ¶ 123; Ex. 1006, 8). And, to avoid unnecessary switching costs, Petitioner contends that one of ordinary skill in the art

would have sought to implement the Agrawal-Brent-Feitelson combination to keep tasks that are included both in the present and new allocations of successive scheduling quanta (CAPs) assigned to their same respective cores. [Ex. 1003 ¶ 124; Ex. 1006 ¶¶ 59–60]. Likewise, it would have been obvious to place each of the activating tasks (***activating program instances***) in the resulting combination to one of the “available cores” (*see supra*,

Element [1.3.iv]) since the other allotted cores would all be occupied by the continuing tasks.

Pet. 57 (citing Ex. 1003 ¶ 124).

### *3. Patent Owner's Contentions*

Patent Owner argues that Petitioner's ground of unpatentability based on Agarwal, Brent, and Feitelson is deficient. *See* Prelim. Resp. 1–12. According to Patent Owner, “[c]laims 10 and 29 both recite ‘monitoring capacity demand indicators . . . expressing a number of instances that the given program has available ***for execution for a succeeding Core Allocation Period (CAP)***.’” *Id.* at 10. As such, Patent Owner argues that “the[se] claims require a ‘capacity demand indicator,’ which expresses a number of instances that the program has available for a ***next*** CAP.” *Id.* However, other than identifying these elements, Patent Owner does not contest Petitioner's challenge to claim 10 other than by relying on its arguments presented for claim 15, as discussed above. *See* Prelim. Resp. 10–12.

### *4. Conclusion for Claim 10*

Having considered the positions of the parties, we conclude that, at this stage of the proceeding, Petitioner has shown a reasonable likelihood that it would prevail in challenging the patentability of independent claim 10. In our view, Petitioner's obviousness approach, on this record, adequately identifies where all the elements of claim 10 are found in the prior art, and, in considering claim 10 as a whole, Petitioner demonstrates adequate reasoning to combine the teachings of Agrawal, Brent, and Feitelson.

Accordingly, for these reasons, and the reasons set forth with respect to claim 15 above, we determine that Petitioner has established a reasonable

likelihood of prevailing on its assertion that claim 10 would have been obvious over the combination of Agrawal, Brent, and Feitelson.

*5. Independent Claim 19*

According to Patent Owner, “[r]ather than reciting ‘capacity demand indicators,’ claim 19 recites the claim term ‘actually materialized demands.’” Prelim. Resp. 13–14.

Petitioner provides an element-by-element analysis of where each element in the challenged claims is disclosed in, or would have been obvious in view of, Agrawal, Brent, and Feitelson. Pet. 59–67. Petitioner also provides a reason why it would have been obvious to modify and combine the teachings of Agrawal, Brent, and Feitelson with a reasonable expectation of success, as proposed by Petitioner. *Id.* Petitioner relies on the testimony of Dr. Chase for evidentiary support. Ex. 1003 ¶¶ 133–149.

In response, Patent Owner argues that “the Petition and Chase declaration point to Agrawal but fail to identify with specificity what in Agrawal constitutes ‘actually materialized demand,’” as claimed. *Id.* at 14 (citing Pet. 30, 61). Patent Owner further argues that, if “the Petition relied on Agrawal’s ‘desire’ as representing ‘actually materialized demand,’ nothing in Agrawal expresses program demand for an upcoming core allocation period” because “Agrawal’s desire is based entirely on historic process utilization, not actually materialized demand.” *Id.* at 15.

At this stage of the proceeding, we do not agree that Petitioner fails to identify adequately what in Agrawal constitutes “actually materialized demand,” as recited by claim 19. To address this element of claim 19, Petitioner asserts

Agrawal thus discloses a first round of allocation for each scheduling quantum in which a “job scheduler” allocates a number of processors among the jobs (*software programs*) so that any actually materialized demands for processors by each of the jobs up to their entitled processor availability  $p_q$  (*entitled shares*) are met.

Pet. 61 (citing Ex. 1003 ¶¶ 136–141; Ex. 1004, 1–2). Relying on the declaration testimony of Dr. Chase, Petitioner concludes that one of ordinary skill in the art “would have found it obvious to define a minimum partition size for each job or class of jobs in the resulting combination to ‘guarantee a certain level of service’ to the jobs.” *Id.* (citing Ex. 1003 ¶ 137). And, to the extent Patent Owner argues that Agrawal does not disclose this element because “Agrawal’s desire is based entirely on historic process utilization, not actually materialized demand” (Prelim. Resp. 15), we do not agree with this argument for the reasons discussed above with respect to claim 15. In this regard, we understand “instantaneous feedback” data to describe the demands for processors based on the processing period that is just about to begin, which is same as the claimed “upcoming” period. *See* Ex. 1004, 2 (“Between quanta  $q - 1$  and  $q$ , the task scheduler determines its job’s *desire*  $d_q$ , which is the number of processors the job wants for quantum  $q$ .”).

Accordingly, for these reasons, and the reasons set forth with respect to claim 15 above, we determine that Petitioner has established a reasonable likelihood of prevailing on its assertion that claim 19 would have been obvious over the combination of Agrawal, Brent, and Feitelson.

#### 6. *Independent Claim 29*

According to Patent Owner claim 29 is similar to claim 10. *See* Prelim. Resp. 10. Patent Owner does not contest Petitioner’s challenge to

claim 29 other than by relying on its arguments presented for claims 10 and 15, as discussed above. *See id.* at 10–12.

Petitioner provides an element-by-element analysis of where each element in the challenged claims is disclosed in, or would have been obvious in view of, Agrawal, Brent, and Feitelson. Pet. 72–73. Petitioner also provides a reason why it would have been obvious to modify and combine the teachings of Agrawal, Brent, and Feitelson with a reasonable expectation of success, as proposed by Petitioner. *Id.* Petitioner also relies on the testimony of Dr. Chase for evidentiary support. *Id.* (citing Ex. 1003 ¶¶ 162–177).

Accordingly, for the reasons set forth with respect to claims 10 and 15 above, and on this preliminary record, we determine that Petitioner has established a reasonable likelihood of prevailing on its assertion that claim 29 would have been obvious over the combination of Agrawal, Brent, and Feitelson.

*7. Dependent Claims 13, 14, 20–22, 24, 25, 27, 32, and 33*

Claims 13, 14, 20–22, 24, 25, 27, 32, and 33 depend directly or indirectly from at least one of claims 10, 19, and 29. Petitioner provides a clause-by-clause analysis of where each element in dependent claims 13, 14, 20–22, 24, 25, 27, 32, and 33 is disclosed in, or would have been obvious in view of, Agrawal, Brent, and Feitelson. Pet. 58–59, 68–73. Throughout Petitioner’s analysis, Petitioner cites and relies on the testimony of Dr. Chase for evidentiary support. *Id.* (citing Ex. 1003 ¶¶ 129–130, 150–163, 174–177).

Patent Owner does not specifically address any of the dependent claims. *See* Prelim. Resp. 1–15.

We find Petitioner’s arguments and evidence in support thereof sufficient at this stage of the proceeding for the reasons explained by Petitioner. *See* Pet. 58–59, 68–73. Accordingly, based on the arguments and evidence on the record before us, we determine, for purposes of this Decision, Petitioner has established a reasonable likelihood of prevailing on dependent claims 13, 14, 20–22, 24, 25, 27, 32, and 33.

### III. DISCRETIONARY DENIAL

Institution of *inter partes* review is discretionary. *See Harmonic*, 815 F. 3d at 1367 (“[T]he PTO is permitted but never compelled to institute an [*inter partes* review (IPR)] proceeding.”); 35 U.S.C. § 314(a).

Patent Owner does not analyze each of the factors set forth in *Apple Inc. v. Fintiv, Inc.*, IPR2020-00019, Paper 11, 5–6 (PTAB Mar. 20, 2020) (precedential) (“*Fintiv*”). Prelim. Resp. 15. Instead, with respect to respect to *Fintiv* Factor 1, Patent Owner simply informs us that Petitioner’s motion to stay the parallel litigation in the United States District Court for the Western District of Washington” was denied. *Id.* (citing Ex. 2002). After the Preliminary Response was filed, however, Patent Owner informs us “that the related matter in the Western District of Washington is now stayed pending resolution of the present *inter partes* review proceeding.” Paper 11, 2.

We have not been informed by the parties that a trial date has been scheduled in the parallel litigation. In light of this development, it is not necessary to analyze the *Fintiv* factors in detail because the parallel litigation has been stayed. *See Fintiv*, Paper 11 at 6 (“A district court stay of the litigation pending resolution of the PTAB trial allays concerns about inefficiency and duplication of efforts. This fact has strongly weighed

against exercising the authority to deny institution under [*NHK Spring Co. v. Intri-Plex Techs., Inc.*, IPR2018-00752, Paper 8 (PTAB Sept. 12, 2018) (precedential, designated May 7, 2019)].”); *Precision Planting, LLC v. Deere & Co.*, IPR2019-01052, Paper 19 at 10 (PTAB Jan. 7, 2020) (finding that the district court stay of the parallel district court case rendered moot the patent owner’s argument for discretionary denial of the petition).

Accordingly, we decline to exercise our discretion to deny institution.

#### IV. CONCLUSION

For the foregoing reasons, we conclude that the information presented in the Petition establishes a reasonable likelihood that Petitioner would prevail in showing that at least one of claims 10, 13–15, 17–22, 24, 25, 27, 29, 32–34, and 37 of the ’833 patent is unpatentable. Accordingly, we institute *inter partes* review of all claims on all grounds raised in the Petition.

Our determination in this Decision is not a final determination on the patentability of any challenged claims and, thus, leaves undecided any factual issues necessary to determine whether sufficient evidence supports Petitioner’s contentions by a preponderance of the evidence in the final written decision. *See TriVascular, Inc. v. Samuels*, 812 F.3d 1056, 1068 (Fed. Cir. 2016) (noting that “there is a significant difference between a petitioner’s burden to establish a ‘reasonable likelihood of success’ at institution, and actually proving invalidity by a preponderance of the evidence at trial”) (quoting 35 U.S.C. § 314(a) and comparing § 316(e)).

#### V. ORDER

In consideration of the foregoing, it is hereby:

ORDERED that, pursuant to 35 U.S.C. § 314(a), an *inter partes* review of claims 10, 13–15, 17–22, 24, 25, 27, 29, 32–34, and 37 of U.S. Patent No. 9,632,833 B2 is instituted with respect to all grounds set forth in the Petition; and

FURTHER ORDERED that, pursuant to 35 U.S.C. § 314(c) and 37 C.F.R. § 42.4, notice is hereby given of the institution of a trial, which will commence on the entry date of this decision.

IPR2022-00527  
Patent 9,632,833 B2

FOR PETITIONER:

Walter Renner  
David Holt  
Nicholas Stephens  
Jennifer Huang  
FISH & RICHARDSON P.C.  
Axf-ptab@fr.com  
Holt2@fr.com  
nstephens@fr.com  
jjh@fr.com

FOR PATENT OWNER:

Scott McKeown  
ROPES & GRAY LLP  
Scott.mckeown@ropesgray.com

W. Cook Alciati  
Michael Dorfman  
GARDELLA GRACE P.A.  
calciati@gardellagrace.com  
mdorfman@gardellagrace.com