

Filed on behalf of: R2 Solutions LLC

By: Carder W. Brooks
Registration No. 75,456
NELSON BUMGARDNER CONROY P.C.
3131 W. 7th St., Suite 3000
Fort Worth, TX 76107
Telephone: (817) 806-3814
Email: carder@nelbum.com

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

AMERICAN AIRLINES, INC., and HILTON DOMESTIC OPERATING
COMPANY, INC.

Petitioners

v.

R2 SOLUTIONS LLC,

Patent Owner.

Case IPR2023-00689

U.S. Patent 8,190,610

PATENT OWNER'S PRELIMINARY RESPONSE

TABLE OF CONTENTS

I.	INTRODUCTION	1
II.	BACKGROUND	2
A.	Overview of the '610 Patent	2
1.	Introduction to MapReduce	3
2.	Shortcomings of Prior Technologies	7
3.	Improvement to MapReduce Described by '610 Patent.....	10
4.	The Claimed Technology of the '610 Patent Improves MapReduce.	14
III.	LEVEL OF ORDINARY SKILL	20
IV.	CLAIM CONSTRUCTION	20
V.	APPLICABLE LEGAL STANDARDS.....	23
A.	Obviousness	24
1.	Claims cannot be found obvious if an element is absent.....	24
2.	A petition must provide articulated reasoning with rational underpinning to combine and/or modify references.....	25
VI.	SUMMARY OF THE PRIOR ART REFERENCES.....	26
A.	Overview of Primary Reference: Pike.....	26
B.	Overview of Secondary References: Chowdhuri	30
VII.	THE PETITION DOES NOT DEMONSTRATE THAT THE CHALLENGED CLAIMS ARE UNPATENTABLE.	32
A.	Grounds 1 and 2: Pike Does Not Teach or Suggest At Least One Limitation of Each Independent Claim, And Combining Pike with Chowdhuri Does Not Resolve This Flaw.	33

1. Pike does not disclose “data groups.”	34
2. Pike fails to disclose delineating “data groups” prior to partitioning....	39
3. Neither reference discloses “intermediate data” that is “identifiable” to a “data group.”	43
4. Neither reference discloses “data groups” with different schemas.	46
a) Pike fails to disclose data groups with different schemas.....	47
b) Pike in combination with Chowdhuri fails to disclose this limitation.....	50
5. Neither reference discloses a key in common between the (1) “data groups”/schema of the input data, and (2) the intermediate data.	53
6. Neither reference discloses processing intermediate data according to its “data group,” or merging the intermediate data based on a key in common between the intermediate data and the “data groups.”.....	56
VIII. THE PETITION SHOULD BE DENIED UNDER 35 U.S.C. § 325(d).....	60
A. Substantially the Same Art and Arguments were Previously Presented to the Office.....	61
B. Petitioner has Failed to Demonstrate that the Office Committed Material Error.	65
IX. PATENT OWNER’S FINAL COMMENTS	66
X. CONCLUSION.....	67

TABLE OF AUTHORITIES

Cases

<i>Advanced Bionics, LLC v. Med-El Elektromedizinische Geräte GmbH</i> , IPR2019-01469 (PTAB Feb. 13, 2020)	<i>passim</i>
<i>CFMT, Inc. v. YieldUp Int’l Corp.</i> , 349 F.3d 1333 (Fed. Cir. 2003)	24
<i>Garmin Int’l, Inc. v. Patent of Cuozzo Speed Techs. LLC</i> , IPR2012-00001, Paper 15 (PTAB Jan. 9, 2013)	24
<i>Graham v. John Deere Co.</i> , 383 U.S. 1 (1966)	24
<i>In re Kahn</i> , 441 F.3d 977 (Fed. Cir. 2006)	25
<i>In re Rijckaert</i> , 9 F.3d 1531 (Fed. Cir. 1993)	24
<i>In re Royka</i> , 490 F.2d 981 (C.C.P.A. 1974)	24
<i>InTouch Techs., Inc. v. VGo Comm’ns., Inc.</i> , 751 F.3d 1327 (Fed. Cir. 2014)	25, 34
<i>KSR Int’l Co. v. Teleflex Inc.</i> , 550 U.S. 398, 418 (2007)	<i>passim</i>
<i>LG Elecs., Inc. v. Cellular Commc’ns Equip. LLC</i> , IPR2016-00197, Paper 7 (PTAB April 29, 2016)	<i>passim</i>
<i>N.V. v. Abbott Labs.</i> , 512 F.3d 1363 (Fed. Cir. 2008)	25

Rules, Regulations, and Statutes

35 U.S.C. § 103	24
35 U.S.C. § 103(a)	24

35 U.S.C. § 325(d)	<i>passim</i>
37 C.F.R. § 42.108(c).....	23

Other Authorities

Office Patent Trial Practice Guide, 77 Fed. Reg. 48,756 (Aug. 14, 2012)	23
--	----

PATENT OWNER’S EXHIBIT LIST

Exhibit	Description
2001	Order of Dismissal, <i>R2 Solutions LLC v. American Airlines, Inc.</i> , Case No. 4:22-cv-00353 (E.D. Tex. June 26, 2023), ECF 72
2002	Fundamentals of MapReduce
2003	MapReduce Design Patterns
2004	Claim Construction Memorandum Opinion and Order, <i>R2 Solutions LLC v. American Airlines, Inc.</i> , Case No. 4:22-cv-00353 (E.D. Tex. April 3, 2023), ECF 58
2005	Google’s MapReduce patent – what does it mean for Hadoop?
2006	Information Disclosure Statement
2007	US6341289
2008	US20060117036A1

I. INTRODUCTION

The Petition should be denied under 35 U.S.C. § 314(a) because it fails to establish any reasonable likelihood that the challenged claims would be found to be unpatentable. Each of the Petition’s grounds have glaring holes, and neither of the cited references teach or suggest key limitations of the challenged claims. The Petition also fails to articulate any adequate motivation to modify or combine the cited references to achieve the claimed inventions. The fact is that U.S. Patent No. 8,190,610 (the “’610 patent”) claims novel and non-obvious enhancements to MapReduce architecture that are nowhere to be found in the prior art and would not be obvious to a POSITA.

The Petition should also be denied under 35 U.S.C. § 325(d). The Petition presents substantially the same art and arguments that were already before the USPTO during prosecution of the ’610 patent. Thus, under *Advanced Bionics*, it was incumbent on Petitioner¹ to demonstrate how the USPTO erred in a way material to

¹ For convenience, co-petitioners American Airlines, Inc. (“AA”) and Hilton Domestic Operating Company (“Hilton”) are referred to collectively as “Petitioner.” The parallel district court action against AA was resolved and ultimately dismissed on June 26, 2023. *See* Ex. 2001. Counsel for AA contacted the Board on July 5, 2023 to request permission for R2 and AA to submit a Joint

patentability. See *Advanced Bionics, LLC v. Med-El Elektromedizinische Geräte GmbH*, IPR2019-01469, Paper 6 at 7 (precedential). But the Petition is silent as to any alleged error committed by the USPTO. The Board should thus exercise discretion under § 325(d) and deny institution.

II. BACKGROUND

A. Overview of the '610 Patent

The '610 patent, granted to Yahoo!, claims a novel MapReduce architecture that can be applied to affect mapping and reducing across data sets comprising data of different schema. Such architecture, adapted to receive and process data from heterogenous sources, greatly expanded the applicability of MapReduce to dramatically improve big data analytics. This improvement was significant at the time of the '610 patent (and remains significant today), enhancing the overall capabilities of computer functionality as it relates to distributed database processing.

Motion to Terminate this proceeding as to AA, and a Joint Request to Maintain Confidentiality and Keep Separate the settlement agreement that is to be filed with the Board as part of the Motion to Terminate. The Motion to Terminate has not yet been submitted. The anticipated Motion to Terminate will not seek to terminate Hilton as a petitioner.

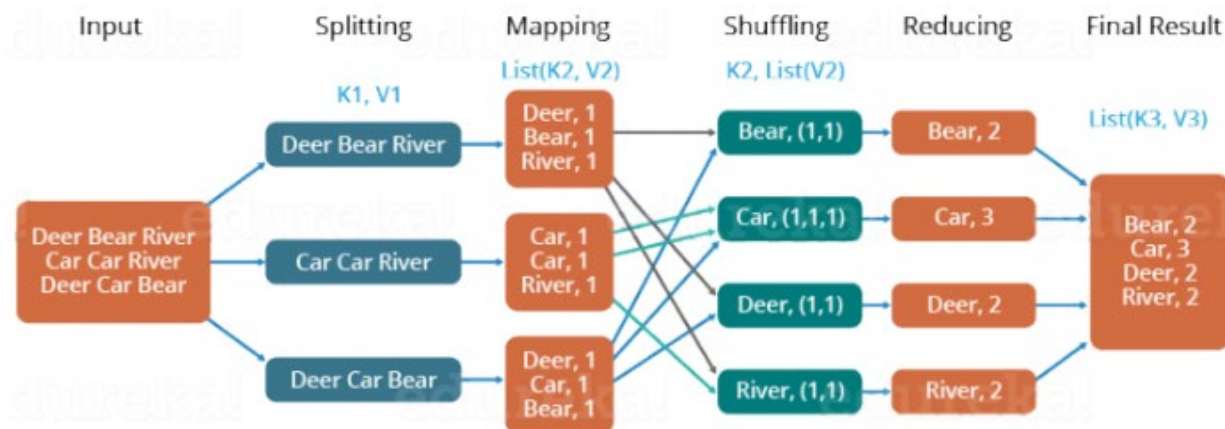
1. *Introduction to MapReduce*

As laid out in the '610 specification, the concept of MapReduce essentially consists of a “‘map’ function [that] maps key-value pairs to new (intermediate) key-value pairs,” and a “‘reduce’ function [that] represents all mapped (intermediate) key-value pairs sharing the same key to a single key-value pair or a list of values.” '610 patent at 1:17-20. At its core, “MapReduce implementations enable the use of massive clusters of commodity computers to provide a simplified programming and execution model for processing large sets of data in parallel.” *Id.* at 9-12.

In a simple example found in U.S. Patent No. 7,590,620 to Pike (“Pike,” Petitioner’s primary reference), MapReduce can be used to “count[] the number of occurrences of each distinct word in a large collection of documents.” Ex. 1002 at 10:36-39. In this example, the map function “outputs a key/value pair for every word w in every document in the collection, where the key/value pair is <w, 1>.” *Id.* at 10:40-43. It is likely that with a voluminous number of documents, each map function “will produce hundreds or thousands of records of the form <word, 1>.” *Id.* at 10:51-52. After the mapping phase, the reduce functions “sort the intermediate data values, merge or otherwise combine sorted intermediate data values having the

same key.” *Id.* at 8:16-19. In this example, the effect is that “[t]he Reduce function simply adds up the count values.” *Id.* at 10:52-53.

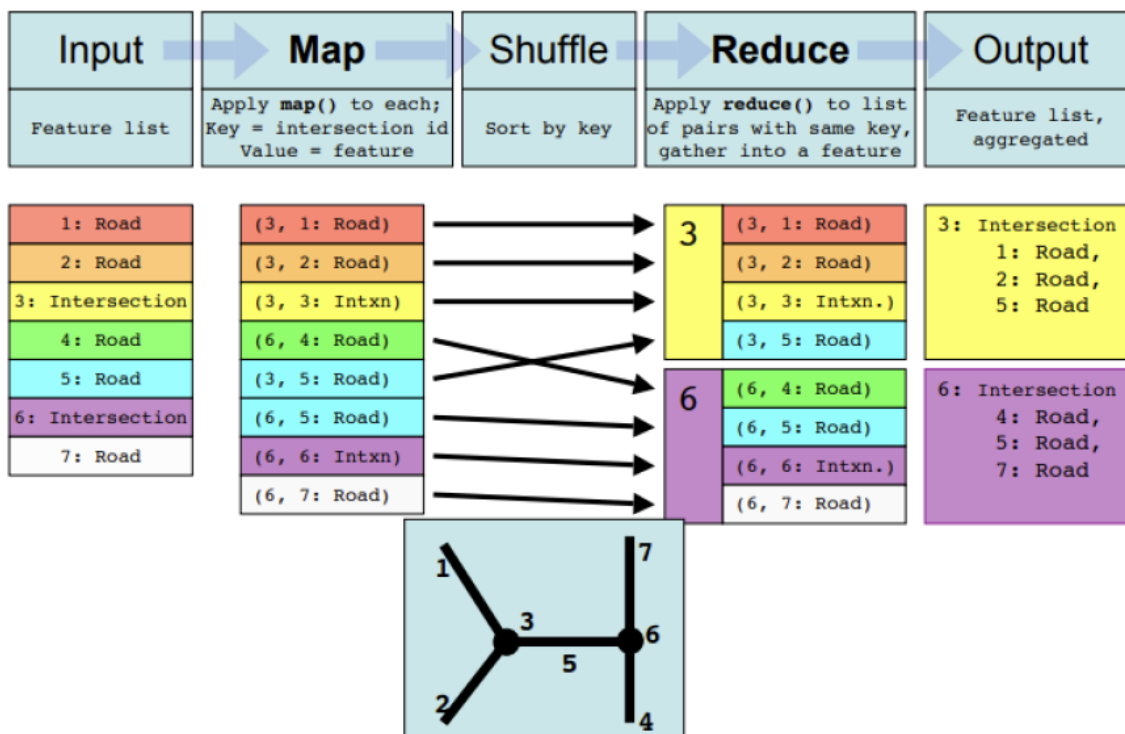
The below graphic illustrates the word count example discussed in Pike:



Ex. 2002 at 5. In this example, the input data consists of a collection of words (Deer, Bear, River, and Car). Applying traditional MapReduce in the “Splitting” step, the input data is split to be supplied to different map processes. Next, in the “Mapping” step, each map function creates lists of key-value pairs. A key-value pair works much like a run-of-the-mill dictionary—the words are the “keys,” and the definitions are the “values.” In the example above, each mapping function generates a list of key-value pairs where the words are the “keys” and the values are “1” to indicate that each word, as a discrete item, occurs one time (e.g., “Deer, 1”). *See* Ex. 1002 at 10:40-43. Obviously, some of the words occur multiple times, and that is accounted for later in the process. At the “Shuffling” step, all key-value pairs having the same key are grouped together to ensure that pairs with the same key are submitted to the same reducers (as mentioned above, reducers reduce based on the same key, also

referred to as a common key). At the “Reducing” step, the reducers can determine how many occurrences of that particular key exist for the received data and can generate another key-value pair, accordingly. *See id.* at 8:16-19. Ultimately, the output is a list of all of the key-value pairs from each reducer that represents the number of times the word appears in the input text (e.g., “Deer, 2” indicating that the word “Deer” appeared twice). *See id.* at 10:52-53; 8:16-19. This is a very simple application of traditional MapReduce.

MapReduce can also be used to accomplish other tasks, such as “joins” or “merges” of data or lists that already exist as key/value pairs. *See id.* at 9:65-66 (“The input data blocks . . . may in some embodiments be treated as key/value pairs . . .”). The below graphic illustrates such a merger:



Ex. 2003 at 8.

Here, the input data is already provided as a list of key-value pairs. In this example, the input data is a list of features in a graph (e.g., roads and intersections) that are numbered; the numbers are the keys and the feature types are the values. In the mapping phase, new keys are assigned that correspond to the intersection that the particular feature participates in (e.g., roads 1, 2, and 5, and intersection 3, all are involved with intersection 3). *See* Ex. 1002 at 7:29-43. Thus, in this example, the key generated in the mapping phase is the intersection ID, and the value is the key-value pair corresponding to the graph feature. The values are shuffled and passed to the reducers based on their common key. *See id.* at 8:16-19. Then, the reducers “reduce” the data to yield the output data, i.e., the reducers can consolidate all of the

features pertaining to a particular common key, such that all of the features participating in intersection 3 fall with intersection 3, and all of the features participating in intersection 6 fall with intersection 6. *See id.*

As is apparent from the above examples, MapReduce offers a way to take large, unwieldy datasets and extract quantifiable and useful information algorithmically. As discussed in the '610 patent, however, in 2006, MapReduce had significant limitations that restricted its usefulness.

2. Shortcomings of Prior Technologies

The '610 patent discloses shortcomings in the prior art and then explains, in detail, the technical way the claimed inventions resolve or overcome those shortcomings. The specification discusses that “conventional MapReduce implementations do not have facility to efficiently process data from heterogeneous sources” and that “it is impractical to perform joins over two relational tables that have different schemas.” '610 patent at 3:9-20. “Schema” can refer to, e.g., “fields” in a particular collection of data. For example, following along with the simple example in the '610 specification at Figure 3, the “All Emp” table (302) and “All Dept” table (304) have different “schema,” because the “All Emp” table has the fields of “DeptID” and “EmpName,” while the “All Dept” table has the fields of “DeptID” and “DeptName”:

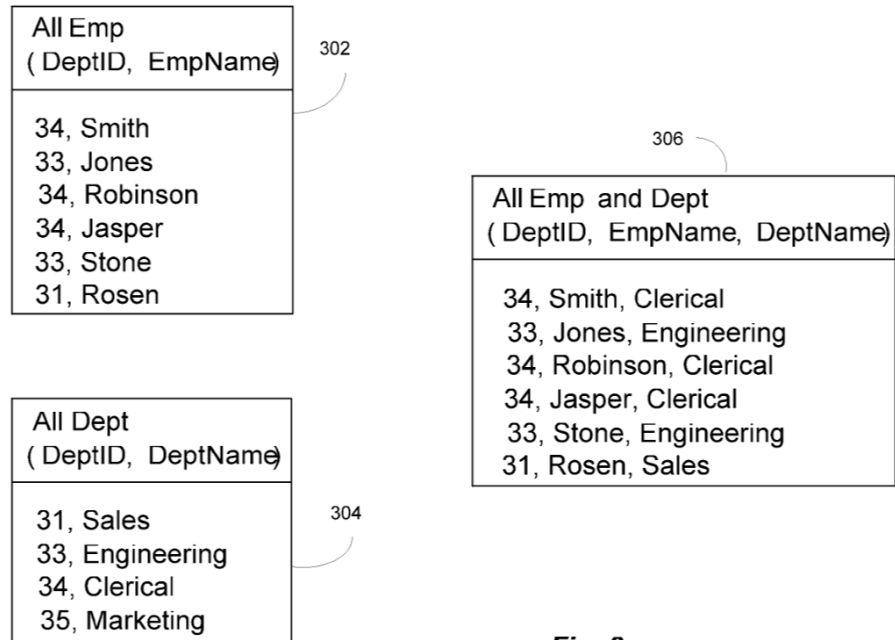
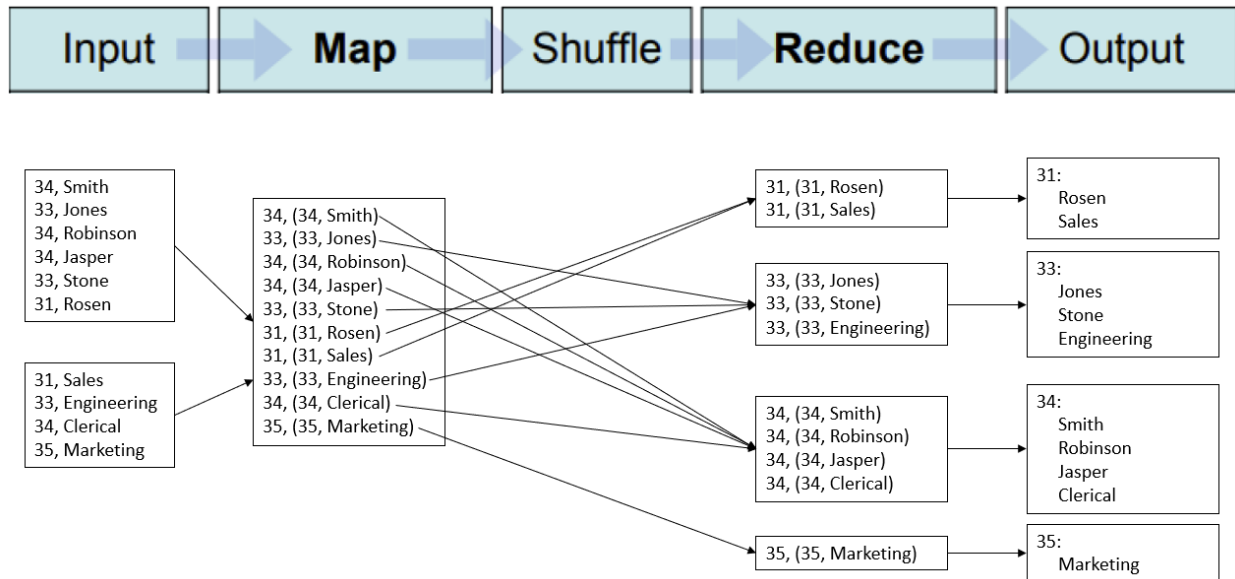


Fig. 3

Id. at FIG. 3; 3:19-20.

With respect to this simple example, the specification explains that it would be inefficient for traditional MapReduce to be implemented to join the “All Emp” table (302) and “All Dept” table (304) to achieve the “All Emp and Dept” table (306). The impracticality of applying traditional MapReduce to join data of different schema is apparent from extrapolation of the traditional MapReduce examples explained above. The below graphic demonstrates the application of traditional MapReduce to the “All Emp” and “All Dept” tables together:



In this example, the “key” for the mapping phase is the “DeptID” (which, mirroring the merger example in Ex. 2003, is a field that each entry shares or participates in) followed by a value of “DeptID, EmpName” or “DeptID, DeptName” (again, mirroring the merger example above). As can be seen here, even though there is a key-in-common amongst the “All Emp” and “All Dept” data sets at the input, there is no joining of the data in any usable way. Instead, the output is a collection of jumbled lists that show both employee names and department names for a single key instead of indicating, e.g., DeptID, employee name, and the associated department name, such as can be seen in the Figure 3 “All Emp and Dept” table (306). For instance, the output list for key 34 in this example contains “Clerical” (a department name) mixed in with “Smith,” “Robinson,” and “Jasper” (employee names)—there is no merger of the two tables to reflect, e.g., “34, Smith,

Clerical” as seen in table 306 of Figure 3. This demonstrates the problem with traditional MapReduce discussed by the ’610 patent.

3. Improvement to MapReduce Described by ’610 Patent

The ’610 patent offers an improvement over traditional MapReduce to specifically address the issues discussed above by using the concept of “data groups.” The specification summarizes these enhancements:

In accordance with an aspect, an input data set is treated as a plurality of grouped sets of key/value pairs, which enhances the utility of the MapReduce programming methodology. Utilizing such grouping, map processing is carried out independently on two or more related datasets (e.g., related by each being characterized by a schema with a key in common). The intermediate results of the map processing (key/value pairs) for a particular key are processed together in a single reduce function by applying a different iterator to intermediate values for each group. Different iterators can be composed inside reduce functions in ways however desired.

Thus, for example, the enhanced MapReduce programming methodology may be easily employed to carry out distributed relational database processing.

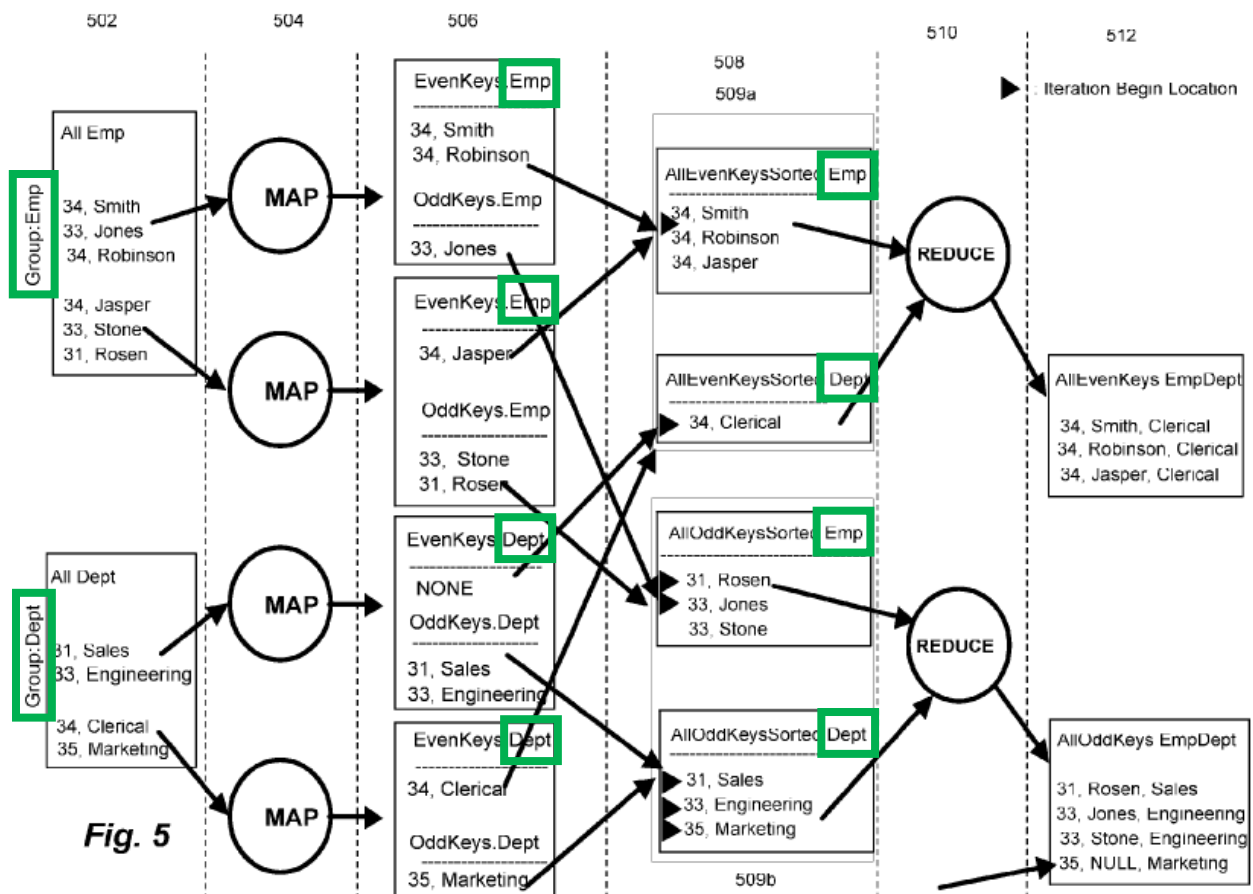
’610 patent at 1:31-44.

The specification goes on to explain how implementation of “data groups,” realizes these improvements:

In general, partitioning the data sets into data groups enables a mechanism to associate (group) identifiers with data sets, map functions and iterators (useable within reduce functions to access intermediate data) and, also, to produce output data sets with (group) identifiers. It is noted that the output group identifiers may differ from the input/intermediate group identifiers.

Id. at 3:58-64.

The '610 patent includes provides a helpful schematic that illustrates how the described and claimed features improve MapReduce:



Id. at FIG. 5 (annotated).

In the above schematic, the input data in portion 502 is shown as the Employee table 302 and the Department table 304 from Figure 3 which share a common key (i.e., DeptID). *See id.* at 8:15-24. Each of these “data groups” include a “mechanism for identifying data from that group”² that is depicted as following the data through the mapping phase 504 and into the reducing phase 510 (i.e., “Emp” and “Dept” emphasized in green). As shown here, the data of each of the “data groups” are partitioned into a “plurality of data partitions” in between portions 502 and 504, and each partition is provided to a “selected one of a plurality of mapping functions” at portion 504. The mapping functions form “intermediate data” at portion 506, and the “intermediate data” is shown as being “identifiable” to its parent “data group” (i.e., via the “Emp” and “Dept” mechanisms). The “intermediate data” also shares the common key (DeptID) with the two “data groups.” After sorting in portion 508, “intermediate data” corresponding to each “data group” is provided to “reduce functions within the portion 510.” *Id.* at 8:34-37. By “processing the intermediate

² As discussed in more detail below, the Eastern District of Texas construed the term “data group” to mean “a group of data and a mechanism for identifying data from that group.” Co-Petitioner AA agreed to this construction in the related patent infringement action, and Petitioner appears to apply this construction in argument. *See, e.g.*, Ex. 1005 at ¶ 87. The Board should likewise adopt that construction.

data for each data group in a manner that is defined to correspond to that data group” in the reducing phase, portion 512 depicts data that is “a merging of the corresponding different intermediate data based on the key in common) (i.e., table 306 in Figure 3).

Thus, the ’610 patent provides a clear technological improvement to existing MapReduce systems by describing and implementing a novel MapReduce architecture where mapping and reduce functions can be applied to data from heterogeneous data sources (i.e., data sources having different schema) to accomplish the merger of heterogeneous data based on a key in common between or among the heterogeneous data. The specification goes so far as to provide specific examples of how a “data group”-oriented MapReduce architecture can be deployed in programming. The ’610 patent shows, for example, that a “data group” identifier can be passed as a parameter to a mapping function and thereby travel through to the resulting intermediate data, followed by a reduce function that employs the “data group” identifier to call particular programming objects (e.g., iterators) associated with specific “data groups”:

```
map(string group, string key, string val) {
    // group: "emp" or "dept"
    // key: DeptID
    // val: LastName for "emp", DeptName for "dept"
    DeptID = key;
    emit_to_group(group, DeptID, val);
}
```

```
reduce(hashtable iterators) {
    // iterators: one iterator for "emp", another for "dept"
    DeptID = iterators.get_key();
    emp_iter = iterators["emp"].get_iter();
    dept_iter = iterators["dept"].get_iter();
    for each LastName in emp_iter do {
        for each DeptName in dept_iter do {
            emit_to_group("emp_dept", DeptID, (LastName,
                DeptName));
        }
    }
}
```

Id. at 6:29-33, 39-48.

Following a similar scheme, another specific embodiment is offered in the context of FIGS. 3-5, again showing how a “data group” identifier (annotated in red) can be indicated to the mapping and/or reduce functions, such that different “data groups” can be mapped differently and/or processed in a way that corresponds to a particular “data group”:

```
map("emp", 34, "Smith") map("dept", 34, "Clerical")  
  
reduce((34, ("emp", ("Smith", "Robinson", "Jasper")), ("dept",  
("Clerical"))))
```

Id. at 7:45, 55, 64-65.

4. *The Claimed Technology of the '610 Patent Improves MapReduce.*

The technological solution described in the specification is embodied, for example, in the independent claims of the '610 patent, reproduced below in a bulleted-list format (for clarity) with particular emphasis on key technological features:

Claim 1:

A method of processing data of a data set over a distributed system, wherein the *data set comprises a plurality of data groups*, the method comprising:

- *partitioning the data of each one of the data groups into a plurality of data partitions* that each have a plurality of key-value pairs and

- *providing each data partition to a selected one of a plurality of mapping functions* that are each user-configurable to
 - independently output a plurality of lists of values for each of a set of keys found in such map function's corresponding data partition to form *corresponding intermediate data for that data group* and *identifiable to that data group*,
 - wherein *the data of a first data group has a different schema than the data of a second data group* and
 - *the data of the first data group is mapped differently than the data of the second data group* so that different lists of values are output for the corresponding different intermediate data,
 - *wherein the different schema and corresponding different intermediate data have a key in common*; and
- reducing the intermediate data for the data groups to at least one output data group, including
- *processing the intermediate data for each data group in a manner that is defined to correspond to that data group*,
- *so as to result in a merging of the corresponding different intermediate data based on the key in common*,
- wherein the mapping and reducing operations are performed by a distributed system.

(emphasis added). Claim 17 echoes the requirements of Claim 1.

Claim 33:

A map-reduce method of processing data from *a plurality of groups having different schema* over a computer system, the method comprising:

- for a first data set having a plurality of first key-value pairs,
 - wherein such *first data set belongs to a first data group* and *the first key-value pairs have a first schema*,
 - *partitioning the first data set into a plurality of data partitions* and
 - providing each data partition to a selected one of a plurality of mapping functions that are each user-configurable to independently output a plurality of lists of values for each of a set of keys found in such map function's corresponding data partition
 - to form *a first intermediate data set having a first set of resulting key-value pairs*;
- for a second data set having a plurality of second key-value pairs,
 - wherein such *second data set belongs to a second data group* and *the second key-value pairs have a second schema*,
 - *partitioning the first data set into a plurality of data partitions* and providing each data partition to a selected one of a plurality of mapping functions
 - to form *a second intermediate data set having a second set of resulting key-value pairs differing from the first set of resulting key-value pairs*,
- wherein *the first schema differs from the second schema*,

- wherein *the first and second schema and the first and second set of resulting key-value pairs have a key in common*; and
- *reducing the first and the second intermediate data set together* so as to form an output data set,
 - wherein the reducing is accomplished by *iterating on at least one key from the first and second intermediate data set*, and
 - *the output data set has a different schema than the first and second schema*,
- wherein the mapping and reducing operations are performed by a distributed system.

(emphasis added). Claim 40 similarly echoes the requirements of Claim 1 and Claim 33.

The independent claims succinctly claim improved MapReduce architecture that embodies the exact technological improvements outlined in the specification. A crucial feature is the implementation of what is referred to as a “data group,” as discussed above. By utilizing “data groups,” a data set with data of different schema can be fed to a collection of mapping functions within a MapReduce architecture to ultimately be reduced and/or merged in spite of the diversity, and this is facilitated by a characteristic of each “data group” (i.e., the “mechanism for identifying data from that group”) of the diverse data set following the data through the mapping phase and into the reducing phase. In this manner, the claimed MapReduce

architecture of is able, in the reducing phase, to selectively employ specialized processing based on the “data group” from which the data being reduced originated. These claimed features enable the MapReduce architecture in the reducing phase to accomplish the merger of intermediate data hailing from different data groups.

For example, Claim 1 describes that “data groups,” each with a different schema, are partitioned, and the resulting partitions are provided to “a selected one of a plurality of mapping functions” such that “the data of the first data group is mapped differently than the data of the second data group.” This requires that the architecture differentiates between “data groups” at the mapping phase to obtain “intermediate data” on a per-“data group” basis, which avoids jumbling of data having different schemas into the same map and thus hindering the reducing phase. After partitions of each of the “data groups” are provided to selected mapping functions, the claim describes that the mapping functions output “intermediate data” for a particular “data group” and that the “different schema [of the data groups] and corresponding different intermediate data have a key in common.” This “key in common” requirement is a necessary feature, because it ultimately serves as the key on which the data of the “data groups” is merged in the reducing phase.

Claim 1 further requires that the “intermediate data” be “identifiable” to its parent “data group” (i.e., via the “mechanism for identifying data from that [data] group”)—this is essential. As revealed later in the claim, the MapReduce

architecture “process[es] the intermediate data for each data group in a manner that is defined to correspond to that data group, so as to result in a merging of the corresponding different intermediate data based on the key in common.” In other words, the reduce functions recognize that received “intermediate data” stems from particular “data groups,” and can therefore employ processing that corresponds to each “data group” to accomplish a combined reducing phase that merges the data (each with a different schema) together. The specification provides helpful context regarding this step, disclosing that, in one embodiment, the reducing phase can employ “data group”-specific iterators to accomplish merger, and this is only possible because of the improved MapReduce architecture’s implementation of “data groups.” *See, e.g.,* ’610 patent at 6:35-7:23. Without “intermediate data” that is identifiable to a particular “data group,” Claim 1’s MapReduce architecture in the reducing phase would not be able to distinguish one type of “intermediate data” from another, which would (in one example) lead to the useless amalgamation of data discussed in Section II.A.2 above.

Hence, the claimed features describe a MapReduce architecture that accounts for different “data group” schema to accomplish a reduction and merger of data having different schema. Vitally, claim limitations ultimately facilitate “a merging of the corresponding different intermediate data based on the key in common,” despite the fact that the “data of a first data group has a different schema than the

data of a second data group.” In sum, the claims’ use of “data groups,” such as exemplified above, enables the MapReduce architecture to achieve merger of heterogenous data. As explained in the specification, such task was ineffective using traditional MapReduce architectures. *Id.* at 3:9-20

III. LEVEL OF ORDINARY SKILL

For the limited purpose of this Preliminary Response, Patent Owner does not contest Petitioner’s definition of a person of ordinary skill in the art, but it reserves the right to do so in the event that trial is instituted.

IV. CLAIM CONSTRUCTION

In R2’s infringement action against AA, the parties agreed to the following constructions for terms of the ’610 patent (*see* Ex. 2004 at 8-9):

<u>Term/Phrase</u>	<u>Agreed Construction</u>
“a plurality of mapping functions that are each user-configurable”	“two or more mapping functions that are each configurable by a user”
“data group”	“a group of data and a mechanism for identifying data from that group”
“a method of processing data of a data set over a distributed system, wherein the data set comprises a plurality of data groups, the method comprising”	Limiting preamble in Claim 1

Judge Mazzant also issued a claim construction order in the case against AA construing several terms of the '610 patent (*see id.* at 10-19):

<u>Term/Phrase</u>	<u>Court's Construction</u>
"identifiable to [that/the] data group"	plain meaning
"independently output a plurality of lists of values"	plain meaning
"the data of the first data group is mapped differently than the data of the second data group"	plain meaning

Judge Mazzant's order in the AA case followed a claim construction order from litigation against other parties that adopted the above constructions and further construed additional terms of the '610 patent (*see Ex. 1007, 13-31*):

<u>Term/Phrase</u>	<u>Court's Construction</u>
"data set"	plain meaning
"data partition"	plain meaning
"including processing the intermediate data for each data group in a manner that is defined to correspond to that data group, so as to result in a merging of the corresponding different	plain meaning

<u>Term/Phrase</u>	<u>Court's Construction</u>
intermediate data based on the key in common”	
“the at least one output data group is a plurality of output data groups”	plain meaning

R2 and Hilton have not yet engaged in claim construction proceedings in their dispute in the Northern District of Texas.

For the purposes of Preliminary Response, Patent Owner submits that the Board should adopt and apply the construction of “data group” identified above. Petitioner does not appear to contest this construction and seemingly applies it in arguing that the prior art teaches “a group of data and a mechanism for identifying data from that group.” Indeed, Petitioner’s expert concedes that “the term ‘data group’ has been construed” as above and later refers to “the claim construction of ‘data group’ as ‘a group of data and a mechanism for identifying data from that group.’” *See, e.g.*, Ex. 1005 at ¶¶ 84, 87. Petitioner and its expert both follow this construction in argument, alleging that that “data groups,” as construed, are taught by the prior art. *See, e.g., id.* at ¶¶ 87 (arguing that “POSITA would have understood that Pike’s input files 302 teach a group of data and a mechanism for identifying data from that group”); Petition at 30 (arguing that the “in input tables are data groups according to the patent because they are identifiable by file name or table name”).

As Petitioner does not contest, this construction of “data groups” is proper in light of the patent itself. As explained by Judge Mazzant in litigation against other parties, the specification discloses that partitioning data “into data groups enables a mechanism to associate (group) identifiers” with the data (’610 patent at 3:58-59), and thus, “the word ‘group’ inherently implies some mechanism to identify the group . . . Finding otherwise would potentially leave the scope of ‘data group’ amorphous, and some construction of the disputed claim language will assist” in understanding the claims. Ex. 1007 at 25 (internal citations omitted). The Board should accordingly construe “data group” to mean “a group of data and a mechanism for identifying data from that group.”

V. APPLICABLE LEGAL STANDARDS

The Board may only grant a petition for *inter partes* review where “the information presented 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.” 35 U.S.C. § 314(a); 37 C.F.R. § 42.108(c). Petitioner bears the burden of showing that this statutory threshold has been met. *See* Office Patent Trial Practice Guide, 77 Fed. Reg. 48,756 (Aug. 14, 2012) (“The Board ... may institute a trial where the petitioner establishes that the standards for instituting the requested trial are met”).

A. Obviousness

Section 103 of the Patent Act provides that “[a] patent may not be obtained . . . 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 said subject matter pertains.” 35 U.S.C. § 103(a). The obviousness analysis requires several threshold inquiries: the level of ordinary skill in the art must be established; the scope and content of the prior art must be determined; and any differences between the prior art and the claims at issue must be ascertained. *Graham v. John Deere Co.*, 383 U.S. 1, 17–18 (1966).

1. *Claims cannot be found obvious if an element is absent.*

If a single element of the claim is absent from the prior art, the claim cannot be considered obvious. *See CFMT, Inc. v. YieldUp Int’l Corp.*, 349 F.3d 1333, 1342 (Fed. Cir. 2003) (“Obviousness requires a suggestion of all limitations in a claim.”) (citing *In re Royka*, 490 F.2d 981, 985 (C.C.P.A. 1974)); *In re Rijckaert*, 9 F.3d 1531, 1534 (Fed. Cir. 1993) (reversing obviousness rejection where prior art did not teach or suggest all claim limitations); *Garmin Int’l, Inc. v. Patent of Cuozzo Speed Techs. LLC*, IPR2012-00001, Paper 15 at 15 (PTAB Jan. 9, 2013) (refusing to institute an *inter partes* review under 35 U.S.C. § 103 where prior art did not disclose all claim limitations).

Additionally, it is improper to use the challenged claim as a roadmap to piece together disparate disclosure using hindsight bias. *See InTouch Techs., Inc. v. VGo Comm'ns., Inc.*, 751 F.3d 1327, 1351 (Fed. Cir. 2014) (reversing obviousness where “[i]t appears that [the expert] relied on the [] patent itself as her roadmap for putting what she referred to as pieces of a ‘jigsaw puzzle’ together.”); *KSR Int’l Co. v. Teleflex Inc.*, 550 U.S. 398, 421 (2007) (“A factfinder should be aware, of course, of the distortion caused by hindsight bias and must be cautious of arguments reliant upon ex post reasoning.”).

2. *A petition must provide articulated reasoning with rational underpinning to combine and/or modify references.*

The conclusion of obviousness based on a combination of references must also be supported with explicit analysis of a reason to combine those references. *KSR Int’l Co.*, 550 U.S. at 418. The Federal Circuit has stated that such reasons must be more than “mere conclusory statements; instead, there must be some articulated reasoning with some rational underpinning to support the legal conclusion of obviousness” as to the particular combination in the claims. *In re Kahn*, 441 F.3d 977, 988 (Fed. Cir. 2006); *see also KSR Int’l Co.*, 550 U.S. at 418 (explaining that the relevant question is whether there was an “apparent reason to combine the known elements in the fashion claimed by the patent at issue”); *Innogenetics, N.V. v. Abbott Labs.*, 512 F.3d 1363, 1374 (Fed. Cir. 2008) (agreeing with the district court’s reasoning that “some kind of motivation must be shown from some source, so that

the jury can understand why a person of ordinary skill would have thought of either combining two or more references or modifying one to achieve the patented method”); *LG Elecs., Inc. v. Cellular Commc 'ns Equip. LLC*, IPR2016-00197, Paper 7 at 7-11 (PTAB April 29, 2016) (faulting a petition’s obviousness analysis for lack of sufficient articulated reasons with rational underpinnings for modifying references to achieve particular elements required in the claims).

VI. SUMMARY OF THE PRIOR ART REFERENCES

A. Overview of Primary Reference: Pike

Pike discloses conventional MapReduce, which is exactly what the ’610 patent seeks to improve upon. Indeed, the parent patent application to Pike (which issued as U.S. Pat. No. 7,650,331) is credited with being a patent on “the principle of distributed MapReduce.” Ex. 2005 at 1. Pike’s Figure 1 depicting its “Large-Scale Data Processing Model” demonstrates this:

Large-Scale Data Processing Model
100

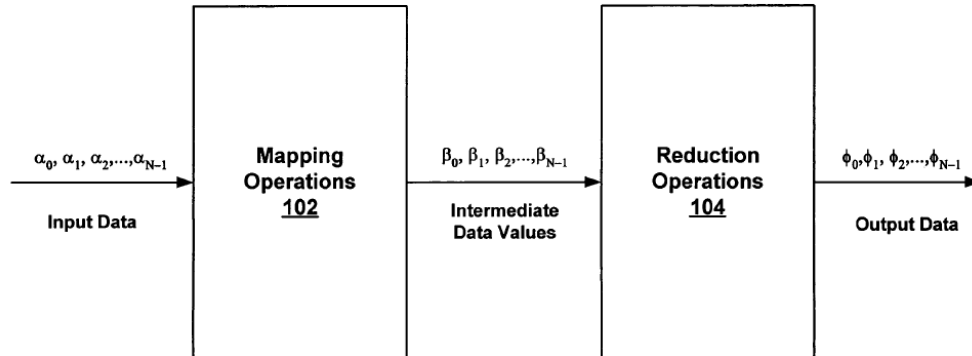


Figure 1

Ex. 1002 at FIG. 1.

Pike explains that “[t]he implementation of large-scale data processing in a parallel and distributed processing environment typically includes the distribution of data and computations among multiple disks and processors to make efficient use of aggregate storage space and computing power.” *Id.* at 1:22-26.” Pike discusses that approaches at that time “fail[ed] to provide support for automatically parallelizing these operations across multiple processors in a distributed and parallel processing environment,” failed to “automatically handle system faults (e.g., processor failures) and I/O scheduling,” and further failed to “efficiently handle the analysis of data records.” *Id.* at 1:30-35. To address these problems, Pike discloses “[a] method and system for analyzing data records [that] includes allocating groups of records to

respective processes of a first plurality of processes executing in parallel. *Id.* at 1:39-41.

The “plurality of processes” in Pike consist generally of “map processes [] to produce a set of intermediate data,” and “reduce processes [] to produce output data” from the “intermediate data.” *Id.* at 3:54-59. Overseeing these processes is a “work queue master” that performs a number of functions, including “determin[ing] how many map tasks to use, how many reduce tasks to use, which processes and processors to use to perform those tasks, where to store the intermediate data and output data, how to respond to any processing failures, and so on.” *Id.* at 4:1-6. Pike’s Figure 2 provides a helpful overview that tracks with the ’610 patent’s discussion of traditional MapReduce. *See* ’610 patent at 1:5-27, 1:65-3:34.

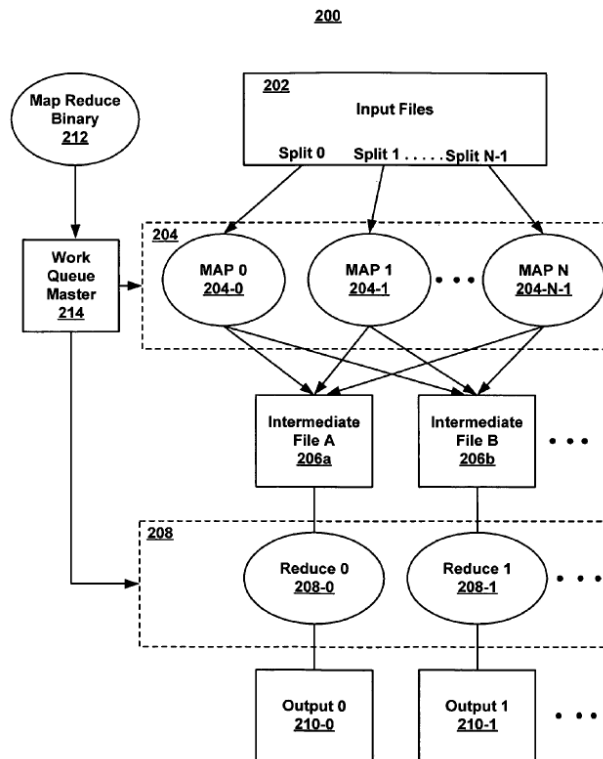


Figure 2

Ex. 1002 at FIG. 2.

In other words, the entire focus of Pike is on solving a different problem than that of the '610 patent—accomplishing conventional MapReduce, which includes the parallelizing of mapping and reducing operations to distribute the processing load. Thus, Pike does not disclose “data groups,” delineating between data based on the “data group” from which the data originated, “intermediate data” that shares a common key with the schemas of the “data groups,” applying MapReduce to data having different schemas to effectuate merger of such data based on a key in common amongst the schemas and the “intermediate data,” or mapping/reducing operations capable of merging data of different schema based on a key in common

via processing on a per-“data group” basis. Put a different way, Pike is concerned with how to accomplish distributed processing, while the ’610 patent is focused on how to improve upon and leverage traditional MapReduce to accomplish merger of heterogenous data. *See* ’610 patent at 3:15-17.

B. Overview of Secondary References: Chowdhuri

Petitioner relies on Chowdhuri as a secondary prior art references. Chowdhuri discloses a “system and methodology for parallel query optimization using semantic-based partitioning.” Ex. 1003 at Abstract. The specification explains that, at that time, “existing partitioning strategies [had] a number of limitations,” that “[c]urrent solutions only work[ed] for a small subset of SQL operations,” and that “[e]xisting partitioning techniques [did] not provide a generalized solution applicable to all database operations, including complex SQL operations such as grouping, joins, distinct operations, unions, and the like.” *Id.* at ¶ 13. Chowdhuri thus sought to create “a solution that provide[ed] the ability to efficiently partition data so that multiple subtasks or operations [could] be performed simultaneously.” *Id.* at ¶ 14.

In Figure 3A, Chowdhuri depicts a “serial operator tree 300 illustrating a serial plan for executing a query,” where “order and customer tables (data 308, 309) are scanned by two separate scan iterators . . . Once these tables are scanned, results are sent to the hash join iterator,” then to a “grouping iterator,” and finally to a “sort iterator.” *Id.* at ¶ 131.

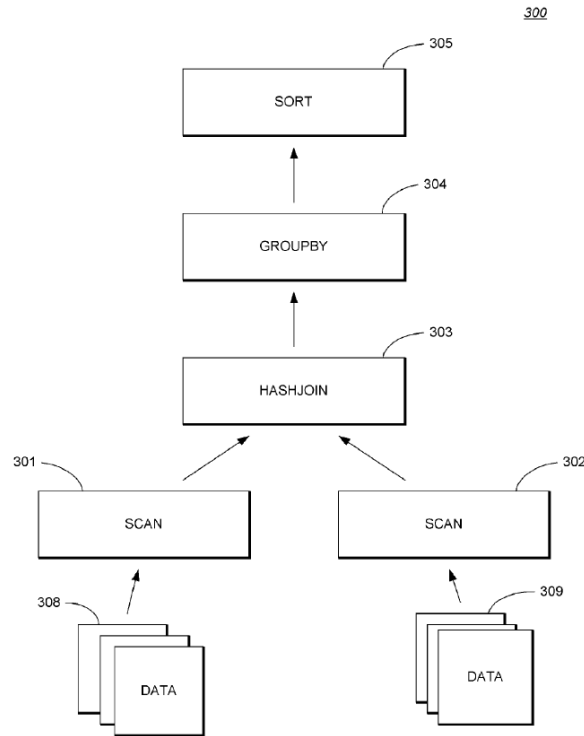


FIG. 3A

Id. at FIG. 3A.

Per Chowdhuri, “[a]n important point about parallelizing a query is to initiate as many iterators as possible and spread them across multiple disk and CPU resources for processing,” and Chowdhuri accomplishes this via an “exchange operator.” *Id.* at ¶ 133. Chowdhuri’s “exchange operator” “permits concurrent execution of complex query evaluation plans in multiple cooperating processes, “allows pipelining of intermediate results from one set of operators (i.e., producer tasks) and initiates another set of operators to process the next request (i.e., consumer tasks),” and “takes the data from the producer tasks and is capable of redistributing data for a different degree of parallelism in its consuming tasks.” *Id.* at ¶ 134.

Thus, much like Pike, Chowdhuri is focused on accomplishing parallel processing, which is not the problem that the '610 patent seeks to solve. Chowdhuri does not disclose “data groups” with different schema, delineating between data based on the “data group” from which the data originated, “intermediate data” that shares a common key with the schemas of the “data groups,” applying MapReduce to data having different schemas to effectuate merger of such data based on a key in common amongst the schemas and the “intermediate data,” or mapping/reducing operations capable of merging data of different schema based on a key in common via processing on a per-“data group” basis.

VII. THE PETITION DOES NOT DEMONSTRATE THAT THE CHALLENGED CLAIMS ARE UNPATENTABLE.

To justify the institution of an *inter partes* review, a petitioner must establish that there is a “reasonable likelihood that the petitioner would prevail with respect to at least 1 of the claims challenged in the petition.” 35 U.S.C. § 314(a). Petitioner has not met its burden of establishing a reasonable likelihood of success at least because the Petition fails to show that each limitation of the independent claims is disclosed or taught by the asserted prior art references of each ground.³

³ Because Petitioner has failed to carry its burden of demonstrating a reasonable likelihood of success with each of the challenged independent claims, Petitioner

Consequently, the Board should deny the Petition and decline to institute the *inter partes* review.

A. Grounds 1 and 2: Pike Does Not Teach or Suggest At Least One Limitation of Each Independent Claim, And Combining Pike with Chowdhuri Does Not Resolve This Flaw.

As discussed above, the '610 patent claims an enhanced MapReduce architecture that accomplishes merger of data having different schema via the use of “data groups,” a feature unique to the '610 patent. The enhancements are reflected as requirements of each independent claim. Claims 1 and 33 are reproduced in Section II.A.4 above, and Claims 17 and 40 mirror the requirements of Claims 1 and 33 respectively.

The crux of Petitioner’s argument is that Pike discloses conventional MapReduce, Chowdhuri discloses tables with different schema, and combining these references teaches all of the limitations of the independent claims. But Petitioner ignores the requirements of the claims. Both Pike and Chowdhuri are silent on the concept of “data groups” and on delineating between data of different schema at all, so neither Pike by itself nor in combination with Chowdhuri render the claims of the '610 patent obvious. Petitioner’s arguments improperly use the

likewise has failed to carry its burden of demonstrating a reasonable likelihood of success with each of the challenged dependent claims.

claims as a roadmap to piece together disparate disclosure using hindsight bias, the Petition should therefore be denied. *See InTouch Techs.*, 751 F.3d at 1351.

1. *Pike does not disclose “data groups.”*⁴

Each of Claims 1, 17, 33, and 40 require “data groups,” and the Petition fails to show that “data groups” are rendered obvious by the prior art. Petitioner first contends that the ’610 patent “admits that this feature [data groups] was conventional.” Petition at 30. Not true. In discussing conventional MapReduce, the ’610 patent explains that “the input data set 102 is partitioned into an arbitrary grouping of seven partitions 102(1) through 102(7).” ’610 patent at 2:31-33. But the “grouping” of partitions with respect to conventional MapReduce is not a “data

⁴ The limitations relevant to this argument are: [1a], [17a], and [40a] (“wherein the data set comprises a plurality of data groups”); [33a] (“A map-reduce method of processing data from a plurality of groups having different schema over a computer system”); [33b] and [40b] (“for a first data set having a plurality of first key-value pairs, wherein such first data set belongs to a first data group and the first key-value pairs have a first schema”); and [33f] and [40f] (“for a second data set having a plurality of second key-value pairs, wherein such second data set belongs to a second data group and the second key-value pairs have a second schema”).

group” because nothing about the “grouping” provides a “mechanism for identifying *data* from that group.” To the contrary, the grouping in traditional MapReduce is disclosed as “arbitrary,” and nowhere is it disclosed that the architecture identifies which “grouping” particular data originated from. *See id.* The ’610 specification never uses the term “data groups” in connection with explaining the prior art, and this term is only employed to explain the claimed inventions. It thus cannot be said that the ’610 patent “admits that this feature was conventional.” Petition at 30.

The discussion of traditional MapReduce in the ’610 specification actually emphasizes the problem that the claimed inventions solve via this unique feature. Immediately after the discussion of conventional MapReduce, the ’610 specification explains that “the conventional MapReduce implementations do not have facility to efficiently process data from heterogeneous sources. For example, it is impractical to perform joins over two relational tables that have different schemas.” ’610 patent at 3:15-20. The patent references FIG. 3 and explains that it “graphically illustrates such a join (without regard for any particular method or architecture for accomplishing such [*sic*] join” that conventional MapReduce could not practically accomplish. *Id.* at 3:20-22.

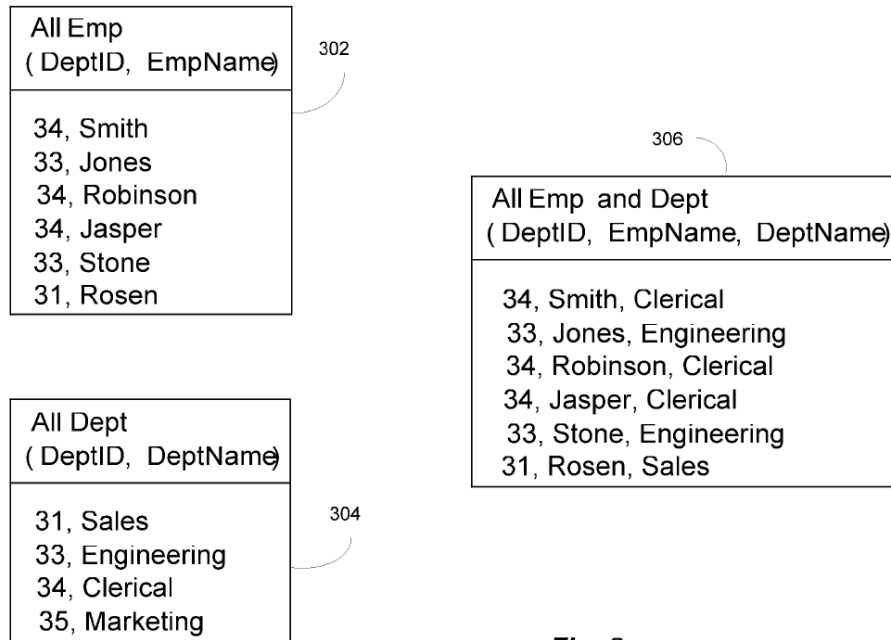


Fig. 3

'610 patent at FIG. 3.

As discussed in Section II.A above, the impracticality of efficiently accomplishing the join stems from the lack of a mechanism to identify which data came from which table. Without the mechanism, traditional MapReduce must treat all of the data as having originated from the same grouping and as having the same schema, and it thus cannot effectuate group-specific processing on the data to accomplish efficient joinder of data having different schema. The '610 patent therefore discloses that “[i]n the *improved* MapReduce architecture . . . the input, intermediate and output data sets are partitioned into a set of data groups . . . [P]artitioning the data sets into data groups enables a mechanism to associate (group) identifiers with data sets, map functions and iterators (useable within reduce functions to access intermediate data) and, also, to produce output data sets with

(group) identifiers.” *Id.* at 3:58-62 (emphasis added). Utilizing such “data groups” was not contemplated by the prior art, much less admitted to be conventional by the ’610 patent.

The only other argument Petitioner offers regarding this limitation is that Pike discloses “data groups.” *See* Petition at 30-31. Relying solely on its expert’s declaration, Petitioner alleges that the input files of Pike “are data groups according to the patent because they are identifiable by file name or table name.” *Id.* at 30. But conflating input files with “data groups” ignores the construction of “data groups” that co-Petitioner AA agreed with and that the Board should adopt.⁵ Indeed, nowhere does Pike teach “a mechanism for identifying *data*” from any particular input file. The file name certainly does not meet this requirement, because the file name *only* identifies the file itself, which is nothing more than the *location* of the data to be processed. Indeed, Pike discloses that a function of the “work queue master” is to determine “where to store” data, i.e., in which files. Ex. 1002 at 4:1-6. The file name

⁵ Additionally, Petitioner’s allegation that a “POSITA would have understood that the ’610 patent’s tables would have been stored as files, like Pike’s input files” is incorrect on its face. Petition at 30-31. Even Pike explains that “the input data may come from sources other than files, such as a database or in-memory data structures.” Ex. 1002 at 9:62-64.

in Pike is not disclosed as identifying the *data* that comes from that file such that the programming architecture can accomplish specific processing based the file name. The only disclosed function of the file name in Pike is to allow a user to specify which files to process, or, in other words, the location of the data to be processed. *See id.* at 4:34-40 (“[T]o perform a specific data processing operation on a set of input files, the only information that must be provided by an application programmer is: information identifying the input file(s), information identifying or specifying the output files to receive output data, and two application-specific data processing operators, hereinafter referred to as `map( )` and `reduce( )`.”) And all that Pike does with a given file name beyond using it to locate which data to process is include it in a “task status table” for the work queue master to “keep[] track of the status of map and reduce tasks.” *Id.* at 55-56. Nowhere does Pike disclose identifying *data* from an input file with the file name, meaning that Pike’s file names cannot be considered “a mechanism for identifying *data* from that [data] group.”

Pike’s usage of file names tracks with the problem that Pike aims to solve and emphasizes why Pike does not disclose “data groups” or render them obvious. Again, the entire purpose of Pike is to accomplish the distribution and parallelization of MapReduce processing. To do this, Pike’s architecture implements a “work queue master” to “assign[] tasks to processes,” and Pike discloses that “multiple processes may be executed by each of the processors in the group of processors that are

available to do the work assigned by the work queue master.” *Id.* at 4:7-11. Thus, the “work queue master” must be capable of effectuating distribution of the processing load, which includes directing input/output to particular nodes and processes and tracking the status of each available process. Pike discloses receiving file names from users to determine which files need to be processed and subsequently distributing processing of the particular files amongst its architecture. But what Pike does *not* disclose is how to deal with data of different schema within the actual data processing steps to accomplish efficient merger of such data. Indeed, Pike is unconcerned with the *type* of data being processed, because Pike is not trying to solve the problem of applying its architecture to data of different schema to merge heterogenous data. Pike cannot be said to disclose this limitation.

*2. Pike fails to disclose delineating “data groups” prior to partitioning.*⁶

The Petition fails to demonstrate that the prior art teaches delineating “data groups” prior to partitioning, as is required for each challenged claim. As an initial

⁶ The limitations relevant to this argument are: [1c] and [17b] (“partitioning the data of each one of the data groups into a plurality of data partitions that each have a plurality of key-value pairs”); [33a] (“A map-reduce method of processing data from a plurality of groups having different schema over a computer system”);

matter, the Petition is wrong that the “’610 patent admits that this feature was conventional by partitioning its input data before mapping.” Petition at 31. As discussed above, “data groups” were not used in conventional MapReduce, so this argument fails for that reason alone. And far from admitting that such feature was conventional, the ’610 patent simply recognizes that in traditional MapReduce, “the input data set . . . is partitioned into an arbitrary grouping of seven partitions.” ’610 patent at 2:31-32. The claims, however, require that the “partitioning” occur on “the data of each one of the data groups” (for Claims 1 and 17) and that the first and second data sets belong to first and second “data groups,” respectively (for Claims 33 and 40), meaning that the “data groups” are delineated prior to the partitioning step. Nowhere does the ’610 patent suggest that an input data set in conventional MapReduce was first segregated into “data groups” prior to partitioning.

[33b] and [40b] (“for a first data set having a plurality of first key-value pairs, wherein such first data set belongs to a first data group and the first key-value pairs have a first schema”); [33c] and [40c] (“partitioning the first data set into a plurality of data partitions”); and [33f] and [40f] (“for a second data set having a plurality of second key-value pairs, wherein such second data set belongs to a second data group and the second key-value pairs have a second schema”).

Further, Petitioner ignores the requirements of the claims in arguing that Pike discloses this feature. While partitioning data was certainly a step in conventional MapReduce and is contemplated by Pike, the claims again require that the “data groups” are delineated prior to the partitioning step. Petitioner’s (incorrect) argument is that Pike’s “input files” are “data groups,” so for Pike to disclose this limitation, each “input file” would have to be partitioned into a “plurality of data partitions.” But this is not what Pike teaches. Instead, Pike just discusses splitting a collection of “input files” based on a “specified or predefined size” of the desired “data blocks” (*id.* at 4:55-56), and Pike teaches that a “data block” can include data from multiple “input files” and that an “input file” can be its own “data block.” *See id.* at 6:59-72 (“In some embodiments, the input files field may specify a portion of an input file (e.g., where the portion comprises a data block) to be processed by the task, or this field may specify portions of two or more input files.”); 4:58-59 (“Alternately, in some embodiments . . . the individual files are the data blocks.”). In other words, Pike teaches both that the “input files” do not have to be partitioned, and that the “data blocks” can be formed from data of multiple “input files.”

This teaching in Pike is contrary to the claims, which require “*partitioning the data of each one of the data groups* into a plurality of data partitions” (for Claims 1 and 17) and that the first and second data sets *belong to first and second “data groups,”* respectively, prior to partitioning (for Claims 33 and 40). The “data

blocks” in Pike are not formed for “each one of the” of the “input files” and do not “belong” to any sort of data group, as is required. Instead, Pike’s “data blocks” can be entire “input files,” parts of “input files,” or parts of multiple “input files,” and are thus not “partitions” formed for “each one of” the “input files” and do not “belong” to any delineated grouping of “input files.” Indeed, Pike splits an amalgamation of “input files” into “data blocks” without any grouping of the files, by file name or otherwise, prior to the split. This is reflected in Pike’s figures 2 and 3, which depict the “input files” as a collection of data that are not grouped in any manner prior to the split.

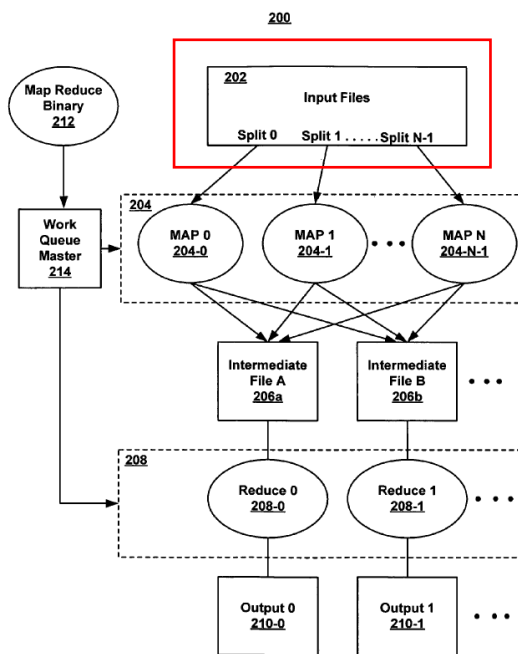


Figure 2

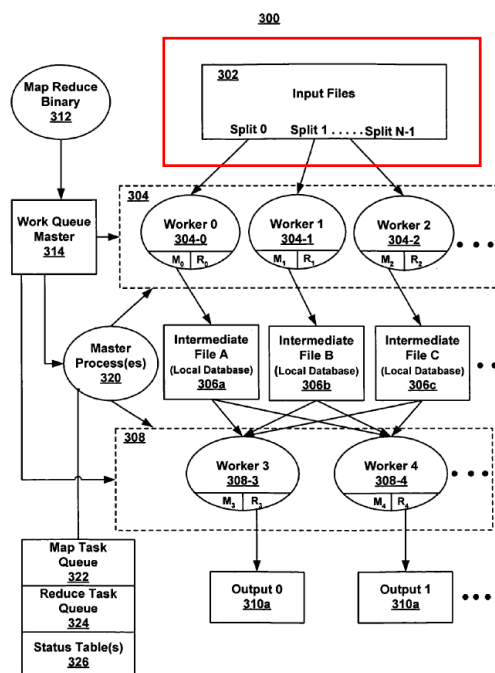


Figure 3

Id. at FIG. 2, FIG. 3 (annotated).

In short, Pike never contemplates any delineation of groups of “input files” or delineation of individual “input files” prior to splitting the files into “data blocks,” emphasizing that (1) the “input files” in Pike cannot be “data groups” within the meaning of the ’610 patent, and (2) Pike does not teach “partitioning the data of each one of the data groups into a plurality of data partitions” (for Claims 1 and 17) or that the first and second data sets belong to first and second “data groups,” respectively, prior to partitioning (for Claims 33 and 40). *Id.* at 4:54-55.

3. *Neither reference discloses “intermediate data” that is “identifiable” to a “data group.”*⁷

Pike fails to disclose or render obvious “intermediate data” that is “identifiable” to the “data group” from which the “intermediate data” came, as is required by each Claims 1 and 17. Petitioner points to the “task status table” in Pike as purportedly disclosing this limitation, alleging that “Pike[] track[s] input files through the mapping and reducing processes” and “generat[es] intermediate data that is identifiable to a particular input file.” Petition at 37. But Petitioner both mischaracterizes Pike and ignores the requirements of the claim language. While

⁷ The limitations relevant to this argument are: [1e] and [17d] (“to independently output a plurality of lists of values for each of a set of keys found in such map function’s corresponding data partition to form corresponding intermediate data for that data group and identifiable to that data group”).

Pike certainly tracks which files are allocated to and generated by particular processes, there is no indication in Pike that the “intermediate *data*” itself is “identifiable” to a given “input file.” To the contrary, Pike’s “task status table” simply reveals that the “work queue master” is aware of (1) which “input files” were handled by which processes, and (2) in which “output files” the resulting “intermediate data” is stored.

Task Status Table					
Task ID	Status	Process	Input Files	Output Files	...
Map0000	Completed	P0000	12340	12341	...
Map0001	Completed	P0001	12344	12345	...
⋮	⋮				
Map0103	In Progress	P0010	12030	12031	...
Map0104	Failed	P0011	10101	12102	...
Map0103b	In Progress	P0010	12030	12031	...
⋮	⋮				
Red0000	In Progress	P0033	12340, ...	14000	...
Red0001	Waiting	-	-	-	...

Figure 7A

Ex. 1002 at FIG. 7A.

In other words, while Pike tracks which “output file” is generated from processing given “input files,” Pike is silent with respect to whether the “intermediate *data*” itself that is stored in the “output file” is identifiable to any original “input file.” In fact, ensuring that “intermediate data” is “identifiable” to a given “input file” is contrary to Pike’s teachings. Pike discloses that a particular process “retrieves data from *one or more input files* (or the one or more input file portions) identified in the input file field and writes the results of the task ID to *one*

or more output files identified in the output file field.” *Id.* at 6:65-7:2. Thus, a given “output file” can contain “intermediate data” stemming from multiple “input files.” But far from teaching that the “intermediate data” in an “output file” (which originates from at least two different “input files” in this example) is “identifiable” to their particular “input files,” Pike teaches that all of the “intermediate data” is amalgamated into an “output file” with a file name that is *different* from that of the “input files.” *See id.* at FIG. 7A (depicting the “output file” name as different from the “input file” name). Indeed, per Pike, the “intermediate data” *loses* its identification with its respective “input files,” because all “intermediate data” generated by a given process, regardless of where it originated from, is simply entered into a single “output file” with a new file name. *See id.*

That Pike does not disclose “intermediate data” that is “identifiable” to a given “input file” is a result of the problem that Pike aims to solve. Again, Pike is not focused on how to apply MapReduce to data having different schema, and so Pike’s architecture has no need to distinguish “intermediate data” based on its original “input file.” But in contrast, the entire point of the ’610 patent is to augment existing MapReduce architecture to efficiently join data of different schema. As discussed below, to accomplish this, the “intermediate data” itself is “identifiable” to its original “data group” so that the “intermediate data” can be processed “in a manner that is defined to correspond to that data group.” *See, e.g.,* ’610 patent at Claim 1.

In short, Pike does not disclose that the “intermediate data” stored in the “output file” retains any sort of identification with the original “input file.” Pike thus cannot be said to disclose these limitations of Claims 1 and 17 or render them obvious.

4. *Neither reference discloses “data groups” with different schemas.*⁸

Neither Pike nor Pike in view of Chowdhuri disclose or render obvious “wherein the data of a first data group has a different schema than the data of a second data group” (as required by Claims 1 and 17) or “wherein the first schema

⁸ The limitations relevant to this argument are: [1f] and [17e] (“wherein the data of a first data group has a different schema than the data of a second data group”); [33a] (“A map-reduce method of processing data from a plurality of groups having different schema over a computer system”); [33b] and [40b] (“for a first data set having a plurality of first key-value pairs, wherein such first data set belongs to a first data group and the first key-value pairs have a first schema”); [33f] and [40f] (“for a second data set having a plurality of second key-value pairs, wherein such second data set belongs to a second data group and the second key-value pairs have a second schema”); [33i] and [40i] (“wherein the first schema differs from the second schema”); and [33l] and [40k] (“the output data set has a different schema than the first and second schema”).

differs from the second schema” (as required by Claims 33 and 40). Indeed, neither reference even uses the term “schema.”

a) Pike fails to disclose data groups with different schemas.

Petitioner first argues that Pike’s disclosure that its “input files . . . can include a variety of data types” discloses data groups with different schemas. *See* Petition at 38. But disclosing that “input files” can be of different types falls short of disclosing this element. Claims 1 and 17 require an input “data set” with a “plurality of data groups,” and that a first and second “data group” of this “data set” have different schemas; Claims 33 and 40 similarly require first and second “data groups” with schemas that are different. Pike simply does not teach this. Indeed, there is *no* requirement in Pike that the “input files” have different schemas, and Pike’s disclosure that the “input files” can be any variety of data types just emphasizes the schema of the data processed by Pike’s architecture is irrelevant. Indeed, as Petitioner points out, a “MapReduce programmer” utilizing Pike’s system “need only know the identity of the input files, where to send output data, and two application-specific data processing operators.” Petition at 39 (internal quotations omitted). Notably absent from this list is the schema of the data being input. Again, this is because Pike is not focused on how to join data of different schema, so the schema is of no moment to Pike.

Additionally, Petitioner repeatedly alleges that it would have been obvious to apply Pike’s mapping and reducing operations to diverse input files and/or to input data with different schema. *See id.* at 39, 40, 41, 42, 43. But this ignores the claimed features of the ’610 patent. *Id.* at 40. The ’610 patent does not claim simply “analyzing input files of a diverse schema”—instead, the claims offer a *specific mechanism* by which MapReduce can be efficiently applied to merge data having different schemas. *Id.* Claims 1 and 17 require that there is an input “data set” comprising a “plurality of data groups” (at least two of which have different schemas) that are each partitioned and processed to form “intermediate data” that is “identifiable” to their particular “data groups.” Similarly, Claims 33 and 40 require delineating input data sets into first and second “data groups” having key-value pairs with schemas that are different. From a high level, Pike plainly does not do this, and it also does not provide any sort of teaching that would lead a POSITA to modify Pike in this fashion. This is apparent from the lack of any discussion in Pike regarding the schema of the input data or any grouping of the input data prior to mapping. It is also apparent from Pike’s lack of disclosure regarding forming “intermediate data” that is “identifiable” to any particular input file to affect reduction that corresponds to any particular input file.

Petitioner is further incorrect that it would be obvious to a POSITA to “maintain a schema-based delineation of data groups in the mapping and reducing”

in the Pike architecture. Petition at 42. Petitioner is using the claims as a roadmap. Pike does not discuss the schema of the data at all, much less delineating between different schema during mapping and reducing operations. And Pike’s discussion of “application-specific operators” does nothing to remedy Pike’s silence regarding the schema of “data groups” or input data sets. *See id.* That operators in Pike can be “application-specific” teaches nothing about utilizing “data groups” with different schema or forming “intermediate data” that is “identifiable” to a particular “data group.” Similarly, Pike’s disclosure of “[a]nalyzing groups of record that include unstructured, ASCII, binary, or other file types” fails to render the claims obvious for the same reason. *See id.* at 43.

Throughout its allegations of obviousness, Petitioner asserts that a POSITA would have several related motivations to modify Pike, including to “apply[] its techniques to input files having different attributes” (*id.* at 40), “to analyz[e] large, diverse collections of documents” (*id.* at 41), “to implement Pike’s mapping and reducing on input files having different schema” (*id.* at 42), and “to accommodate large collections of diverse data sets” (*id.* at 43, 44). *See also* Ex. 1005 at ¶¶ 107-115. None of these motivations are adequate. The proper question regarding a POSITA’s motivation is not whether there was a “general motivation” to apply MapReduce to diverse data sets—it is whether a POSITA would have been motivated to modify Pike to arrive *to achieve the “particular element[s]” of the*

claims of the '610 patent. *See LG Elecs.*, IPR2016-00197, Paper 7 at 8; *see also KSR Int'l Co.*, 550 U.S. at 418 (explaining that the relevant question is whether there was an “apparent reason to combine the known elements in the fashion claimed by the patent at issue”). Pike never discusses schema, delineating between input data based on schema, or forming intermediate data identifiable to a schema-based grouping of input data. Pike also never suggests that such functionality would or should be implemented. No motivation existed for a POSITA to augment Pike in this fashion.

b) Pike in combination with Chowdhuri fails to disclose this limitation.

Petitioner’s reliance on Chowdhuri to fill the holes in Pike is also misplaced. Just like Pike, Chowdhuri never uses the term “schema,” and further does not disclose “data groups” having different schemas. Petitioner alleges that the order and customer tables in Chowdhuri are “data groups” and that these tables have different schema. *See id.* at 44-45. While the order and customer tables in Chowdhuri may have different schema, they are not “data groups”—Chowdhuri discloses no “mechanism for identifying data” from these tables. Indeed, Petitioner did not allege that Chowdhuri teaches “data groups;” Petitioner relied solely on Pike as teaching that limitation. *See id.* at 30-31. And while Petitioner argues here that Chowdhuri’s order and customer tables teach “data groups” with different schema, Petitioner just ignores the agreed-to construction of “data groups” and fails to explain how either of Chowdhuri’s order and customer tables constitute “a group of data and a

mechanism for identifying data from that group.” Chowdhuri teaches no such mechanism (and Petitioner does not allege otherwise).

Additionally, going far beyond the limitation at hand, Petitioner spends several pages attempting to explain that Chowdhuri discloses a MapReduce implementation that a POSITA would be motivated to combine with Pike to render the claims obvious. *See* Petition at 44-48. This discussion culminates in a conclusion that lays bare the deficiencies in Petitioner’s arguments. Petitioner states: “Chowdhuri also expressly discloses what Pike suggests: performing these [mapping and reducing] operations on data having different schema.” *Id.* at 49-50. This isn’t what is claimed by the ’610 patent. The claims do not describe simply applying MapReduce to data having different schema—to the contrary, the claims describe a specific architecture utilizing “data groups” that is engineered “efficiently process data from heterogeneous sources.” *See* ’610 patent at 3:15-17. Petitioner imagines that a POSITA would have a “reasonable expectation of success” in achieving the claims if Chowdhuri were combined with Pike. Petition at 49. But if Chowdhuri’s customer and order tables were simply inserted into Pike’s architecture, the output would simply be an amalgamation of entries with no coherent merger—the exact problem the ’610 patent aims to solve.

Finally, fatal to Petitioner’s combination of Pike with Chowdhuri is the lack of any articulation of an adequate motivation to combine the references. Petitioner’s

only allegation in this regard is that a POSITA “would have been motivated to make this combination to accommodate large collections of diverse data sets.” Petition at 50; *see also* Ex. 1005 at ¶ 125. Again, recitation of a “general motivation” to allow MapReduce to be applied to diverse data is plainly insufficient. *See LG Elecs.*, IPR2016-00197, Paper 7 at 8; *see also KSR Int’l Co.*, 550 U.S. at 418. Indeed, just as in *LG Electronics*, Petitioner alleges that Pike teaches all of the limitations of the independent claims and fails to explain “why a person of ordinary skill would turn to” Chowdhuri for “teaching of an element that, according to Petitioner, is already taught” by Pike. *Id.* The combination that Petitioner puts forth “does not address the particular element for which Petitioner is relying” on Chowdhuri, and the Petition “does not explain why a person of ordinary skill in the art would find it advantageous to specifically look to” Chowdhuri for its teaching of tables with different schema. *Id.* Petitioner’s argument should fail for this reason alone.

Additionally, with respect to Claims 33 and 40, neither reference can be said to disclose that an “output data set” from a reduce function “has a different schema than the first and second schema” of the input sets of key-value pairs, because neither reference discusses “schema” at all, discloses what the schemas of the input sets are, or discloses what the schemas of the output sets are.

5. *Neither reference discloses a key in common between the (1) “data groups”/schema of the input data, and (2) the intermediate data.*⁹

Petitioner fails to establish that the prior art discloses or renders obvious a key in common between (1) “data groups” (for Claims 1 and 17) or schema of the input data (for Claims 33 and 40), and (2) the intermediate data. Petitioner first alleges that Pike’s disclosure of a reduce function that “merges or otherwise combines the sorted intermediate data values having the same key” meets these requirements. Petition at 52. But in addition to ignoring the requirement that the “data groups” or input data sets have “different schema,” this argument also ignores that “the different schema” of the “data groups”/input data sets, *as well as* the corresponding intermediate data, must all have “a key in common.” Pike does not disclose this. Instead, Pike just discloses that the collections of “intermediate data” all have the same key. Nowhere does Pike disclose that the “input files” (which Petitioner alleges are “data groups”) have a key in common with each other, or with the “intermediate

⁹ The limitations relevant to this argument are: [1h] and [17g] (“wherein the different schema and corresponding different intermediate data have a key in common”); and [33j] and [40l] (“wherein the first and second schema and the first and second set of resulting key-value pairs have a key in common”).

data” that is generated by processing the “input files.” It thus cannot be said that Pike discloses this limitation or renders it obvious.

Pike discloses nothing different from traditional MapReduce. *See, e.g.*, ’610 patent at 2:49-51. Indeed, in traditional MapReduce, the entire purpose of the mapping functions was to *generate* common keys such that the reduce function could reduce the data based on that key. This is demonstrated by Pike’s discussion of using a “map-reduce operation for counting the number of occurrence of each distinct word in a large collection of documents.” Ex. 1002 at 10:37-39. Pike discloses that the key/value pair generated by the map functions would be “<w, 1>,” with the “w” corresponding to an identified word and acting as the key. *Id.* at 10:39-53. The reduce function then “adds up the count values” based on the generated, common keys, i.e., the identified words. *Id.* In other words, Pike discloses generating intermediate data with common keys created via mapping functions, and then reducing the data based on the generated common keys. But nowhere does Pike disclose that “input files” share a common key with one another or with the “intermediate data” prior or subsequent to the mapping phase.

Petitioner’s reliance on Chowdhuri for teaching this limitation is similarly misplaced. Chowdhuri does not disclose key/value pairs *at all*, so it cannot disclose that the schemas of the “data groups”/input data sets and the “intermediate data” “have a key in common.” Further, Petitioner alleges that Chowdhuri’s “order and

customer tables are data groups,” its “scanning iterators” are equivalent to map functions, its “hash join, GroupBy, and sort iterators” are equivalent to reduce functions, and its “results” generated by the “scanning iterators” are the equivalent of “intermediate data.” Petition at 45-47. Thus, under Petitioner’s argument, Chowdhuri would have to disclose that the order and customer tables (purported “data groups” with different schema), as well as the “results” generated by the “scanning iterators” (purported “intermediate data”), all share a common key. This is not the case. While Chowdhuri discloses that the order and customer tables both have a “state” field, Chowdhuri is silent regarding whether the “results” from scanning share also have a “state” field. *See* Ex. 1003 at ¶¶ 126-127, 131, 136. Chowdhuri is also silent regarding whether the “state” field is a key at all. Chowdhuri thus does not teach this feature.

Additionally, Petitioner fails, in both the Petition and its expert declaration, to allege any motivation to combine Chowdhuri with Pike to achieve the claim limitations. Indeed, Petitioner never explains why a POSITA would look to Chowdhuri for its teaching of “data groups” sharing a common key with each other and with “intermediate data.” *See KSR Int’l Co.*, 550 U.S. at 418 (explaining that the relevant question is whether there was an “apparent reason to combine the known elements in the fashion claimed by the patent at issue”); *see also LG Elecs.*, Paper 7

at 8. Petitioner’s argument on this point is thus facially deficient, and the Petition fails for this reason alone.

6. *Neither reference discloses processing intermediate data according to its “data group,” or merging the intermediate data based on a key in common between the intermediate data and the “data groups.”*¹⁰

Neither Pike nor Chowdhuri disclose “processing the intermediate for each data group in a manner that is defined to correspond to that data group . . . so as to result in a merging of the corresponding different intermediate data based on the key in common,” as is required by Claims 1 and 17. As discussed above, neither Pike nor Chowdhuri disclose “data groups” that have a “key in common” with each other and the “intermediate data,” so neither reference teaches “merging” based on such a common key. Further, neither Pike nor Chowdhuri disclose that processing “intermediate data” in the reduce phase occurs “in a manner that is defined to correspond” to the “data groups” from which the “intermediate data” originated.

¹⁰ The limitations relevant to this argument are: [1i] and [17h] (“[reducing/reduce] the intermediate data for the data groups to at least one output data group, including processing the intermediate data for each data group in a manner that is defined to correspond to that data group, so as to result in a merging of the corresponding different intermediate data based on the key in common”).

Petitioner's arguments on this feature are almost non-existent, reflecting the holes in Petitioner's references. Again, Petitioner alleges that "input files" in Pike are "data groups." So for Pike to disclose "processing the intermediate data for each data group in a manner that is defined to correspond to that data group," Pike would have to show that in the reduce phase, the "intermediate data" is processed "in a manner that is defined to correspond" to the "input file" from which the "intermediate data" came from. But this is not what Pike teaches. Indeed, per Petitioner, Pike just teaches "tracking each reduce operation associated with an input file." Petition at 54. In other words, all that Pike teaches is tracking which reduce task is assigned to which file. But tracking which operations handle which files is not the same as *processing* the data "*in a manner that is defined to correspond*" to the original input file. Indeed, *nowhere* does Pike disclose that the original input file defines how "intermediate data" from that file is processed, and Pike thus does not disclose the claimed feature.

Pike's lack of disclosure on this point further emphasizes why Pike does not disclose "data groups." Again, an essential feature of the claims is to generate "intermediate data" that is "identifiable" to a particular "data group" so that in the reducing phase, the reduce functions can process the "intermediate data" on a per-"data group" basis such that the data can be "merg[ed] . . . based on the key in common." And Pike *never* discloses that any reduce operators take the original file

into consideration when processing any “intermediate data,” which emphasizes that an “input file” cannot be a “data group,” and a “file name” cannot be such a “mechanism for identifying data” from the “data group.”

Petitioner next points generally to Chowdhuri as disclosing this element, alleging that Chowdhuri’s discussion of Figures 3A and 3B teaches “reducing ‘differently’ or employing different reduce functions . . . specific to each input file” because Chowdhuri “join[s], group[s], and/or sort[s] this intermediate data based on customer or order state.” *See* Petition at 55. This is wrong. As an initial matter, a join, group, or sort of data “based on customer or order state” is not the same thing as “reducing ‘differently’ or employing different reduce functions,” and Petitioner fails to offer any explanation or analysis of how these are equivalent. *Id.* They are not. Chowdhuri’s disclosure of joining, grouping, or sorting data “based on customer or order state” amounts to nothing more than disclosing using a common key to join data, which is exactly what traditional MapReduce does. This reveals nothing of *how* the data is reduced, differently or otherwise, and Chowdhuri simply does not disclose “different reduce functions” as Petitioner alleges. *Id.*

Further, Chowdhuri does not teach processing “intermediate data” “in a manner that is defined to correspond” to the “data group” from which the “intermediate data” came. Again, Petitioner alleges that Chowdhuri’s “order and customer tables are data groups,” its “scanning iterators” are equivalent to map

functions, its “hash join, GroupBy, and sort iterators” are equivalent to reduce functions, and its “results” generated by the “scanning iterators” are the equivalent of “intermediate data.” Petition at 45-47. So to teach “processing the intermediate data for each data group in a manner that is defined to correspond to that data group,” Chowdhuri would have to teach that during the hash join/GroupBy/sort stage (purported “reduce” phase), the results (purported “intermediate data”) generated by scanning the customer and order tables (purported “data groups”) are processed “in a manner defined to correspond” to the customer or order table. Chowdhuri simply does not disclose this. None of the sections of Chowdhuri cited by Petitioner (¶¶ 126-127, 131, 135) disclose that the hash join, GroupBy, or sort operators process the results of the scanning operator in a manner that at all corresponds to which table the results were generated from. To the contrary, all that Chowdhuri says is that “results [from both tables] are sent to the hash join iterator,” “[t]he joined rows are then sent to the grouping iterator (GroupBy),” the “results from the grouping (GroupBy) iterator . . . are passed to the sort iterator,” and the “sort iterator . . . sorts the results.” Ex. 1003 at ¶ 131. Absent from this disclosure is any suggestion that the “results” of the scanning operators are somehow processed “in a manner that is defined to correspond” to the original table from which the results originated. Chowdhuri thus cannot be said to disclose this element.

Finally, Petitioner fails again to articulate an adequate motivation to combine Chowdhuri with Pike to arrive at the specific claimed features. Indeed, Petitioner just baldly alleges that a POSITA would have been motivated to make the combination without explaining what that motivation is. So yet again, Petitioner's argument is fatally flawed, the Petition fails for this reason alone. *See KSR Int'l Co.*, 550 U.S. at 418; *see also LG Elecs.*, Paper 7 at 8.

VIII. THE PETITION SHOULD BE DENIED UNDER 35 U.S.C. § 325(d).

The Director has discretion to deny institution of an *inter partes* review proceeding under 35 U.S.C. § 325(d) when “the same or substantially the same prior art or arguments previously were presented to the Office.” *Advanced Bionics*, Paper 6 at 7. The Board applies a two-part framework under § 325(d): “(1) whether the same or substantially the same art previously was presented to the Office or whether the same or substantially the same arguments previously were presented to the Office; and (2) if either condition of the first part of the framework is satisfied, ***whether the petitioner has demonstrated*** that the Office erred in a manner material to the patentability of challenged claims.”¹¹ *Advanced Bionics*, Paper 6 at 8 (emphasis added).

¹¹ Art that is previously presented includes “art made of record by the Examiner, and art provided to the Office by an applicant, such as on an Information Disclosure

A. Substantially the Same Art and Arguments were Previously Presented to the Office.

The prior art asserted by Petitioner is substantially the same as art previously presented to the Office. Just like in *Advanced Bionics*, Petitioner here relies on references that, while not themselves before the Office during prosecution, nonetheless disclose substantially the same information as references cited and analyzed during prosecution.

The Petition relies on Pike for both grounds, arguing that Pike renders the claims obvious by itself and/or in combination with Chowdhuri. *See* Petition at 13. Petitioner describes Pike as disclosing a “‘large-scale data processing model 100’ for MapReduce, which includes mapping and reduction operations in a map phase 102 and reduce phrase [*sic*] 104, respectively.” *Id.* at 24 (citing Ex. 1002 at 2:12-39). Petitioner also points to Figure 1 and explains that Pike discloses that, over a distributed system, mapping operations are applied to input data to generate intermediate data, and that reduce operations reduce the intermediate data to provide

Statement (IDS), in the prosecution history of the challenged patent.” *Advanced Bionics*, Paper 6 at 8. If the first prong of factor (1) is met (i.e. substantially the same art was previously presented), then no analysis is needed regarding whether the same arguments were previously presented to the Office. *Id.* at 20.

output data. *Id.* In short, Petitioner explains that Pike discloses conventional MapReduce.

The relevant disclosure in Pike is substantially similar, if not identical, to art that was before the USPTO during prosecution. To explain conventional MapReduce and demonstrate its shortcomings, the '610 patent thoroughly discusses a 2004 paper by Jeffrey Dean and Sanjay Ghemawat (Ex. 1009) and a related presentation (Ex. 1011) that each cover traditional MapReduce. *See* '610 patent at 1:6-27; 1:66-3:17. The '610 patent then goes on to explain how the disclosed inventions are different and improve upon traditional MapReduce architecture. *See, e.g., id.* at 3:15-64. In addition to being discussed in the specification, the paper was included in an IDS and listed as a reference cited in the patent. *See* Ex. 2006 at 2. As pointed out by Petitioner, Figures 1 and 2 of the '610 patent were copied from the Dean/Ghemawat presentation. *See* Petition at 14, n. 14. There can thus be no doubt that the Dean/Ghemawat paper and presentation were front and center during prosecution, and ***Petitioner admits that Pike discloses the same thing as these references.*** Indeed, Petitioner alleges that a “POSITA would have recognized that Pike’s mapping and reducing operations correspond to the map and reduce functions in the Dean and Ghemawat paper (EX 1009).” Petition at 26. This is unsurprising, considering that both Dean and Ghemawat are named co-inventors with Robert Pike on the Pike reference.

Patent Owner agrees with Petitioner that Pike corresponds to the Dean/Ghemawat references—they all disclose the same thing, namely, conventional MapReduce. The similarities are glaring. For instance, Figure 3 in Pike is strikingly similar to Figure 1 in the Dean/Ghemawat paper, which is emphasized by the colored annotations below:

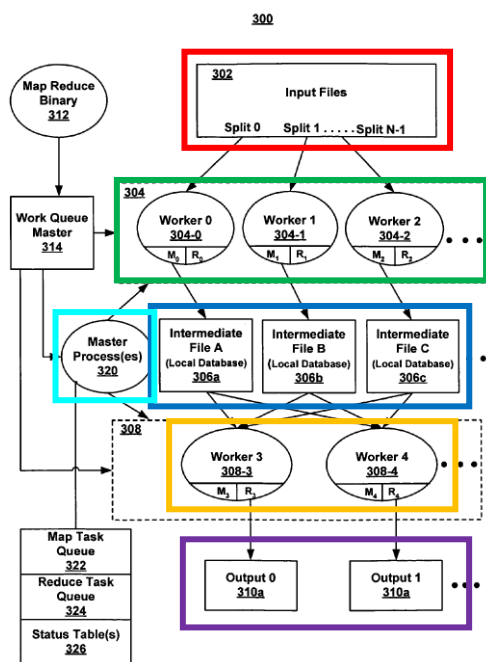


Figure 3

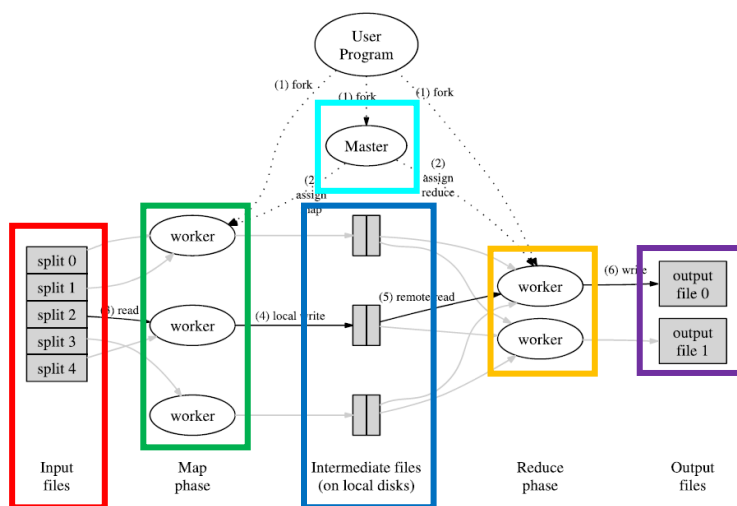


Figure 1: Execution overview

Ex. 1002 at FIG. 3 (annotated)

Ex. 1009 at 3 (annotated)

Additionally, Dean/Ghemawat explains that the user provides “the names of the input and output files” (Ex. 1009 at 2); Pike discloses the same thing (*see* Petition at 30-31, 35-37). Dean/Ghemawat discusses “partitioning the input data into a set of *M splits*” to be “processed in parallel by different machines” (Ex. 1009 at 3-4); Pike discloses the same thing (*see* Petition at 31-32). Dean/Ghemawat discusses a “master” that stores in a data structure the “state (*idle, in-progress, or completed*)”

for “each map and reduce task” and “the locations and sizes” of the intermediate files, and further that input/output files are tracked by the master for the particular tasks (Ex. 1009 at 4); Pike discloses the same thing (*see* Petition at 35-37). In short, *every* feature that Petitioner points to in Pike that purportedly renders the claimed inventions obvious is also disclosed in Dean/Ghemawat.

Chowdhuri is also substantially similar to the art that was considered by the USPTO, and Petitioner’s arguments mirror those of the USPTO that the applicant overcame during prosecution. Petitioner points to Chowdhuri as purportedly disclosing tables with different schema (*see* Petition at 44-50); U.S. Pat. No. 6,341,289 to Burroughs (Ex. 2007) was cited by the USPTO during prosecution as allegedly teaching the same feature. *See* Ex. 1004 at 150. And, unlike Chowdhuri, Burroughs actually uses the term “schema.” *See generally* Ex. 2007. Further, the examiner there put forth the exact same argument that Petitioner is making here, namely, that combining the different schemas in Burroughs with a MapReduce implementation renders the claims obvious. *See* Ex. 1004 at 150. That is precisely what Petitioner has alleged here, only Petitioner has substituted Chowdhuri for Burrows. *See* Petition at 44-50. U.S. App. Pub. No. 2006/0117036 to Cruanes (Ex. 2008) was also cited by the USPTO as allegedly teaching data groups with different schemas. *See* Ex. 1004 at 194-195. And just like Chowdhuri, Cruanes is focused on

performing hash joins of data in parallel—Chowdhuri discloses nothing on this point that was not already considered by the USPTO. *See* Ex. 2008 at ¶¶ 6—18.

Petitioner also relies on Chowdhuri as purportedly teaching processing “intermediate data” “in a manner that is defined to correspond” to its parent “data group” because Chowdhuri discloses joining, grouping, and/or sorting “intermediate data based on customer or order state.” Petition at 55. The USPTO put forth an identical argument as to Cruanes (*see* Ex. 1004 at 230), alleging that Cruanes teaches joining tables based on a matching “deptID” value (*see* Ex. 2008 at ¶ 5). Applicant overcame this argument in prosecution, and the claims were allowed. In summary, every feature of Chowdhuri that Petitioner points to was disclosed in the prior art analyzed by the USPTO, and Chowdhuri is thus substantially similar to art that was before the USPTO during prosecution.

B. Petitioner has Failed to Demonstrate that the Office Committed Material Error.

Because substantially the same art was already presented to the Office, the Board should determine “whether the petitioner has demonstrated that the Office erred in a manner material to the patentability of challenged claims.” *Advanced Bionics*, Paper 6 at 8 (emphasis added). In *Advanced Bionics*, the petitioner presented arguments that the examiner erred during prosecution in evaluating prior art, referring to specific technical aspects and claim limitations that the petitioner asserted the examiner failed to properly analyze and research. *See id.* at 20. Even

though the petitioner there specifically pointed out how it thought the examiner erred and cited additional references that potentially disclosed relevant technical aspects, the Board nonetheless concluded that the petitioner was not persuasive in showing that “the Examiner materially erred in determining the patentability of the challenged claims.” *Id.* at 22.

Here, Petitioner fails to demonstrate that “the Office erred in a manner material to patentability of the challenged claims.” *Id.* at 8. In fact, ***Petitioner does not assert that the Office erred at all.*** Unlike in *Advanced Bionics*, where the Board found that petitioner failed to persuasively demonstrate error when the petitioner actually presented arguments, ***Petitioner here has not even attempted to demonstrate error.*** See *id.* Because substantially the same information was previously before the Office, and because Petitioner has failed to demonstrate that the Office committed material error, the Board should deny institution under § 325(d).

IX. PATENT OWNER’S FINAL COMMENTS

Patent Owner does not concede the legitimacy of any arguments in the Petition that are not specifically addressed herein, and expressly reserves the right to rebut any such arguments in its Patent Owner Response if *inter partes* review is instituted. Similarly, Patent Owner also does not concede any underlying contentions in the Petition and reserves the right to rebut them later. Additionally, Patent Owner is not

limited to the arguments presented here in this Preliminary Response, but expressly reserves the right to raise further arguments, including claim construction arguments, not presented in this Preliminary Response.

X. CONCLUSION

For the foregoing reasons, Patent Owner respectfully requests that the Board decline to institute *inter partes* review of the '610 patent.

Dated: July 11, 2023

By: /s/Carder W. Brooks
Carder W. Brooks
Registration No. 75,456
NELSON BUMGARDNER
CONROY P.C.
3131 W. 7th St.
Fort Worth, TX 76107
Telephone: (817) 806-3814
Email: carder@nelbum.com
Attorney for Patent Owner

CERTIFICATE OF COMPLIANCE WITH WORD COUNT

Pursuant to 37 C.F.R. § 42.24(d), I certify that this Preliminary Response complies with the type-volume limits of 37 C.F.R. § 42.24(b)(1) because it contains 13,885 words, excluding the parts of this Patent Owner's Preliminary Response that are exempted by 37 C.F.R. § 42.24(a), according to the word processing system used to prepare this Patent Owner's Preliminary Response.

Dated: July 11, 2023

By: /s/Carder W. Brooks
Carder W. Brooks
Registration No. 75,456
NELSON BUMGARDNER
CONROY P.C.
3131 W. 7th St.
Fort Worth, TX 76107
Telephone: (817) 806-3814
Email: carder@nelbum.com
Attorney for Patent Owner

CERTIFICATE OF SERVICE

The undersigned certifies that the foregoing Patent Owner's Preliminary Response was served on July 11, 2023, to Lead and Back-up Counsel for Petitioner at the service address provided in Petitioner's Mandatory Notices:

Email: jcampbell@McKoolSmith.com
 rchabaneix@McKoolSmith.com
 speters@McKoolSmith.com
 dfinkelson@mcguirewoods.com
 ksmith@mcguirewoods.com

Dated: July 11, 2023

By: /s/Carder W. Brooks
 Carder W. Brooks
 Registration No. 75,456
 NELSON BUMGARDNER
 CONROY P.C.
 3131 W. 7th St.
 Fort Worth, TX 76107
 Telephone: (817) 806-3814
 Email: carder@nelbum.com

 Attorney for Patent Owner