

CLAIMS

1. A method for dynamic job performance in secure multiparty computation (SMPC), the method comprising:
 - receiving an SMPC query that indicates a processing job to be performed on a data input;
 - determining a query type of the SMPC query and parameters of the data input;
 - splitting the data input to generate a plurality of partial data inputs, based on the parameters and the query type;
 - generating a first plurality of jobs to perform on the plurality of partial data inputs;
 - pushing each of the first plurality of jobs into a worker queue;
 - performing each of the first plurality of jobs in the worker queue on a respective partial data input of the plurality of partial data inputs to determine a plurality of data outputs;
 - in response to determining that the query type indicates that the plurality of data outputs should be joined, generating a second plurality of jobs to perform on the plurality of data outputs;
 - pushing each of the second plurality of jobs into the worker queue; and
 - performing each of the second plurality of jobs in the worker queue on a respective partial data output of the plurality of data outputs to determine a combined result of the processing job.
2. The method of claim 1, wherein the SMPC query comprises respective data inputs from a plurality of nodes and wherein the SMPC query is received by all of the plurality of nodes from a client controller.
3. The method of claim 2, wherein the node comprises a pool of worker processes that perform the plurality of jobs pushed into the worker queue, wherein the plurality of jobs comprises at least the first plurality of jobs and the second plurality of jobs, and wherein a respective worker process of the pool performs a respective job popped from the worker queue until all of the plurality of jobs are complete.

4. The method of claim 3, wherein the pool of worker processes communicate with a respective pool of worker processes on each of the plurality of nodes, and wherein results associated with SMPC queries from each of the nodes are exchanged using tokens that identify a respective result, a respective node, and the SMPC query.
5. The method of claim 3, further comprising:
 - increasing an amount of worker processes in the pool of worker processes in response to receiving the SMPC query; and
 - decreasing the amount of worker processes in the pool in response to determining the combined result.
6. The method of claim 5, wherein the amount of worker processes is at least one of: proportional to an amount of jobs in the first plurality of jobs and, less than a maximum amount of workers.
7. The method of claim 5, wherein increasing and decreasing the amount of worker processes in a worker pool is based on at least one of: required computation, time of day, date, financial costs, power consumption, and available network bandwidth.
8. The method of claim 1, wherein the first plurality of jobs and the second plurality of jobs may comprise at least one of:
 - (1) sorting,
 - (2) filtering,
 - (3) mapping,
 - (4) reducing, and
 - (5) performing a mathematical operation.
9. The method of claim 1, wherein the combined result is stored as one of a secret share, a hash value, or a plaintext value.

10. The method of claim 1, further comprising:
caching the SMPC query and the combined result;
receiving another SMPC query;
in response to determining that the another SMPC query matches the SMPC query,
outputting the cached combined result.
11. The method of claim 1, further comprising:
caching the plurality of partial data inputs and the first plurality of jobs;
receiving another SMPC query comprising another data input;
splitting the another data input into a third plurality of jobs such that the third plurality of jobs comprises the first plurality of jobs and a plurality of another partial data inputs comprises the plurality of partial data inputs;
determining that a partial result of performing the third plurality of jobs on the plurality of another partial data inputs is the plurality of data outputs from the first plurality of jobs; and
performing a remainder of the third plurality of jobs on a remainder of the plurality of another partial data inputs.
12. The method of claim 1, further comprising:
subsequent to receiving the SMPC query, receiving another SMPC query; and
appending another data input of the another SMPC query to the data input of the SMPC query.
13. The method of claim 12, further comprising pushing jobs associated with the SMPC query and the another SMPC query in parallel.
14. The method of claim 1, wherein the data input is a secret share, received from a client controller, that is encoded or has additional encoding such that contents of the data input are securely searchable.

15. The method of claim 1, wherein generating the first plurality of jobs to perform on the plurality of partial data inputs further comprises automatically compiling at least one program to execute on the plurality of partial data inputs, wherein the at least one program is cached.

16. The method of claim 1, wherein the data input is one of:

- (1) an output from a secure function;
- (2) data stored in an internal database;
- (3) data retrieved from an external database; and
- (4) included as a parameter of the SMPC query.

17. The method of claim 1, further comprising:

outputting, to a client controller, a progress of job completion, wherein completion of each respective job of the first plurality of jobs and the second plurality of jobs is monitored and displayed as the progress.

18. The method of claim 1, further comprising:

verifying each respective result of the plurality of data outputs subsequent to performing each of the first plurality of jobs; and

in response to detecting an error in a respective message authentication code associated with a respective result, re-performing a respective job of the first plurality of jobs on a respective partial data input of the plurality of partial data inputs, wherein the respective job and the respective partial data input correspond to the respective result with the error.

19. A system for dynamic job performance in secure multiparty computation (SMPC), the system comprising:

a hardware processor configured to:

- receive an SMPC query that indicates a processing job to be performed on a data input;
- determine a query type of the SMPC query and parameters of the data input;
- split the data input to generate a plurality of partial data inputs, based on the parameters and the query type;
- generate a first plurality of jobs to perform on the plurality of partial data inputs;

push each of the first plurality of jobs into a worker queue;
perform each of the first plurality of jobs in the worker queue on a respective partial data input of the plurality of partial data inputs to determine a plurality of data outputs;
in response to determining that the query type indicates that the plurality of data outputs should be joined, generate a second plurality of jobs to perform on the plurality of data outputs;
push each of the second plurality of jobs into the worker queue; and
perform each of the second plurality of jobs in the worker queue on a respective partial data output of the plurality of data outputs to determine a combined result of the processing job.

20. A non-transitory computer readable medium storing thereon computer executable instructions for dynamic job performance in secure multiparty computation (SMPC), including instructions for:

receiving an SMPC query that indicates a processing job to be performed on a data input;
determining a query type of the SMPC query and parameters of the data input;
splitting the data input to generate a plurality of partial data inputs, based on the parameters and the query type;
generating a first plurality of jobs to perform on the plurality of partial data inputs;
pushing each of the first plurality of jobs into a worker queue;
performing each of the first plurality of jobs in the worker queue on a respective partial data input of the plurality of partial data inputs to determine a plurality of data outputs;
in response to determining that the query type indicates that the plurality of data outputs should be joined, generating a second plurality of jobs to perform on the plurality of data outputs;
pushing each of the second plurality of jobs into the worker queue; and
performing each of the second plurality of jobs in the worker queue on a respective partial data output of the plurality of data outputs to determine a combined result of the processing job.

CLAIMS

1. A method for dynamic job performance in secure multiparty computation (SMPC), the method comprising:

receiving an SMPC query that indicates a processing job to be performed on respective data inputs from a plurality of nodes, wherein the SMPC query is received by all of the plurality of nodes from a client controller;

determining a query type of the SMPC query and parameters of the data input;

splitting each of the respective data inputs to generate a plurality of partial data inputs, based on the parameters and the query type;

generating a first plurality of jobs based on the processing job to perform on the plurality of partial data inputs;

pushing each of the first plurality of jobs into a worker queue;

performing, by the plurality of nodes, each of the first plurality of jobs in the worker queue on a respective partial data input of the plurality of partial data inputs to determine a plurality of partial data outputs;

in response to determining that the query type indicates that the plurality of data outputs should be joined, generating a second plurality of jobs based on the processing job to perform on the plurality of partial data outputs;

pushing each of the second plurality of jobs into the worker queue; and

performing, by the plurality of nodes, each of the second plurality of jobs in the worker queue on a respective partial data output of the plurality of partial data outputs to determine a combined result of the processing job.

2. The method of claim 1, wherein each of the plurality of nodes comprises a pool of worker processes that perform a plurality of jobs pushed into the worker queue, wherein the plurality of jobs comprises at least the first plurality of jobs and the second plurality of jobs, and wherein a respective worker process of the pool performs a respective job popped from the worker queue until all of the plurality of jobs are complete.

3. The method of claim 2, wherein the pool of worker processes communicate with a respective pool of worker processes on each of the plurality of nodes, and wherein results associated with SMPC queries from each of the nodes are exchanged using tokens that identify a respective result, a respective node, and the SMPC query.

4. The method of claim 2, further comprising:
increasing an amount of worker processes in the pool of worker processes in response to receiving the SMPC query; and
decreasing the amount of worker processes in the pool in response to determining the combined result.

5. The method of claim 4, wherein the amount of worker processes is at least one of: proportional to an amount of jobs in the first plurality of jobs and, less than a maximum amount of workers.

6. The method of claim 4, wherein increasing and decreasing the amount of worker processes in a worker pool is based on at least one of: required computation, time of day, date, financial costs, power consumption, and available network bandwidth.

7. The method of claim 1, wherein the first plurality of jobs and the second plurality of jobs may comprise at least one of:

- (1) sorting,
- (2) filtering,
- (3) mapping,
- (4) reducing, and
- (5) performing a mathematical operation.

8. The method of claim 1, wherein the combined result is stored as one of a secret share, a hash value, or a plaintext value.

9. The method of claim 1, further comprising:
caching the SMPC query and the combined result;
receiving another SMPC query;
in response to determining that the another SMPC query matches the SMPC query,
outputting the cached combined result.

10. The method of claim 1, further comprising:
caching the plurality of partial data inputs and the first plurality of jobs;
receiving another SMPC query comprising another data input;
splitting the another data input into a third plurality of jobs such that the third plurality of jobs comprises the first plurality of jobs and a plurality of another partial data inputs comprises the plurality of partial data inputs;
determining that a partial result of performing the third plurality of jobs on the plurality of another partial data inputs is the plurality of data outputs from the first plurality of jobs; and
performing a remainder of the third plurality of jobs on a remainder of the plurality of another partial data inputs.

11. The method of claim 1, further comprising:
subsequent to receiving the SMPC query, receiving another SMPC query; and
appending another data input of the another SMPC query to the data input of the SMPC query.

12. The method of claim 11, further comprising pushing jobs associated with the SMPC query and the another SMPC query in parallel.

13. The method of claim 1, wherein the data input is a secret share, received from a client controller, that is encoded or has additional encoding such that contents of the data input are securely searchable.

14. The method of claim 1, wherein generating the first plurality of jobs to perform on the plurality of partial data inputs further comprises automatically compiling at least one program to execute on the plurality of partial data inputs, wherein the at least one program is cached.

15. The method of claim 1, wherein the data input is one of:

- (1) an output from a secure function;
- (2) data stored in an internal database;
- (3) data retrieved from an external database; and
- (4) included as a parameter of the SMPC query.

16. The method of claim 1, further comprising:

outputting, to a client controller, a progress of job completion, wherein completion of each respective job of the first plurality of jobs and the second plurality of jobs is monitored and displayed as the progress.

17. The method of claim 1, further comprising:

verifying each respective result of the plurality of data outputs subsequent to performing each of the first plurality of jobs; and

in response to detecting an error in a respective message authentication code associated with a respective result, re-performing a respective job of the first plurality of jobs on a respective partial data input of the plurality of partial data inputs, wherein the respective job and the respective partial data input correspond to the respective result with the error.

18. A system for dynamic job performance in secure multiparty computation (SMPC), the system comprising:

at least one hardware processor configured to:

receive an SMPC query that indicates a processing job to be performed on respective data inputs from a plurality of nodes, wherein the SMPC query is received by all of the plurality of nodes from a client controller;

determine a query type of the SMPC query and parameters of the data input;

split each of the respective data inputs to generate a plurality of partial data inputs, based on the parameters and the query type;

generate a first plurality of jobs based on the processing job to perform on the plurality of partial data inputs;

push each of the first plurality of jobs into a worker queue;

perform, by the plurality of nodes, each of the first plurality of jobs in the worker queue on a respective partial data input of the plurality of partial data inputs to determine a plurality of partial data outputs;

in response to determining that the query type indicates that the plurality of data outputs should be joined, generate a second plurality of jobs based on the processing job to perform on the plurality of partial data outputs;

push each of the second plurality of jobs into the worker queue; and

perform, by the plurality of nodes, each of the second plurality of jobs in the worker queue on a respective partial data output of the plurality of partial data outputs to determine a combined result of the processing job.

19. A non-transitory computer readable medium storing thereon computer executable instructions for dynamic job performance in secure multiparty computation (SMPC), including instructions for:

receiving an SMPC query that indicates a processing job to be performed on respective data inputs from a plurality of nodes, wherein the SMPC query is received by all of the plurality of nodes from a client controller;

determining a query type of the SMPC query and parameters of the data input;

splitting each of the respective data inputs to generate a plurality of partial data inputs, based on the parameters and the query type;

generating a first plurality of jobs based on the processing job to perform on the plurality of partial data inputs;

pushing each of the first plurality of jobs into a worker queue;

performing, by the plurality of nodes, each of the first plurality of jobs in the worker queue on a respective partial data input of the plurality of partial data inputs to determine a plurality of partial data outputs;

in response to determining that the query type indicates that the plurality of data outputs should be joined, generating a second plurality of jobs based on the processing job to perform on the plurality of partial data outputs;

pushing each of the second plurality of jobs into the worker queue; and

performing, by the plurality of nodes, each of the second plurality of jobs in the worker queue on a respective partial data output of the plurality of partial data outputs to determine a combined result of the processing job.