

What is claimed is:

1. A method implemented by a data processing system for enabling an application in a container image to execute across host systems by assigning instances of the container image to the host systems and enabling a portion of the application in an instance on a host system to output data to another portion of the application in another instance on another host system, the method including:

accessing the container image that includes the first application and a second application, wherein the second application causes a communication between the data processing system and a host system for executing an instance of the container image;

determining, by the data processing system, the number of parallel executions of the first application;

generating a plurality of instances of the container image in accordance with the number of parallel executions determined, with an instance including the first and second applications;

configuring a first instance to execute a first portion of the first application on a first host system;

causing each of the plurality of configured instances to execute on one or more of the host systems;

receiving, from the second application of a second instance on a second host system, a communication;

based on the communication received from the second application of the second instance on the second host system, transmitting, by the data processing system to the first instance on the first host system, information for the first instance to locate a second portion of the application of the second instance on the second host system to receive an output from the first portion; and

based on the transmitted information to locate the second portion, causing the first portion of the application of the first instance on the first host system to transmit its output data to the second portion of the application of the second instance on the second host system.

2. The method of claim 1, further including accessing a specification that specifies a number of parallel executions of the given module.

3. The method of claim 1, wherein the container image is a first container image, and wherein the method further includes:

accessing from a hardware storage device an instance of a second container image, wherein the second container image includes the first application and specifies an amount of parallelism for each module in the second application; and

storing the second container image on an execution system.

4. The method of claim 3, further including:

transmitting, from the instance of the second container image, a request to an interface included in the execution system to generate the plurality of instances of the container image in accordance with the number of parallel executions determined.

5. The method of claim 1, further including:

assigning each of the plurality of configured instances to one or more of the host systems.

6. The method of claim 5, wherein assigning is dynamically performed by each configured instance being assigned at run-time and the assignments not being pre-determined.

7. The method of claim 6, wherein the first application is a dataflow graph with a plurality of components, further including:

for each component, generating a plurality of instances of the container image in accordance with a number of parallel executions determined for that component; and

dynamically running a dataflow graph on multiple nodes to obtain dynamic levels of parallelism by:

for each component,

dynamically assigning a generated instance to one or more nodes of the host systems; and

causing each of the plurality of assigned instances to execute on the one or more nodes of the host systems.

8. The method of claim 1, wherein the communication between the data processing system and the one or more of the host systems executing that configured instance includes:

transmitting, by the given module, monitor data to the second application, and passing, by the second application, the monitor data to the data processing system, wherein the monitor data is configured to be used by the data processing system to track when the given module has completed execution for the data processing system to instruct the another module to execute.

9. The method of claim 1, wherein the given module is a first module, wherein the generated plurality of instances is a plurality of first instances, and wherein the operations further include:

determining, by the data processing system, a number of parallel executions of a second module of the first application;

for the second module, generating a plurality of second instances of the container image in accordance with the number of parallel executions determined for the second module, with each second instance including the first and second applications;

for each second instance, configuring that second instance to execute the second module of the first application;

causing each of the plurality of configured, second instances to execute on one or more of the host systems; and

retrieving, from a local data store, an address of one of the second instances; and

in accordance with the retrieved address, providing, from one of the first instances to the one of the second instances, output from the one of the first instance.

10. A data processing system for enabling an application in a container image to execute across host systems by assigning instances of the container image to the host systems and enabling a portion of the application in an instance on a host system to output data to another portion of the application in another instance on another host system, the data processing system including:

one or more processing devices; and

one or more machine-readable hardware storage devices storing instructions that are executable by the one or more processing devices to perform operations including:

accessing the container image that includes the first application and a second application, wherein the second application causes a communication between the data processing system and a host system for executing an instance of the container image;

determining, by the data processing system, the number of parallel executions of the first application;

generating a plurality of instances of the container image in accordance with the number of parallel executions determined, with an instance including the first and second applications;

configuring a first instance to execute a first portion of the first application on a first host system;

causing each of the plurality of configured instances to execute on one or more of the host systems;

receiving, from the second application of a second instance on a second host system, a communication;

based on the communication received from the second application of the second instance on the second host system, transmitting, by the data processing system to the first instance on the first host system, information for the first instance to locate a second portion of the application of the second instance on the second host system to receive an output from the first portion; and

based on the transmitted information to locate the second portion, causing the first portion of the application of the first instance on the first host system to transmit its output data to the second portion of the application of the second instance on the second host system.

11. The system of claim 10, wherein the operations further include accessing a specification that specifies a number of parallel executions of a given module.

12. The system of claim 10, wherein the container image is a first container image, and wherein the operations further include:

accessing from a hardware storage device an instance of a second container image, wherein the second container image includes the first application and specifies an amount of parallelism for each module in the second application; and

storing the second container image on an execution system.

13. The system of claim 12, wherein the operations further include:

transmitting, from the instance of the second container image, a request to an interface included in the execution system to generate the plurality of instances of the container image in accordance with the number of parallel executions determined.

14. The system of claim 10, wherein the operations further include:
assigning each of the plurality of configured instances to one or more of the host systems.

15. The system of claim 14, wherein assigning is dynamically performed by each configured instance being assigned at run-time and the assignments not being pre-determined.

16. The system of claim 15, wherein the first application is a dataflow graph with a plurality of components, wherein the operations further include:

for each component, generating a plurality of instances of the container image in accordance with a number of parallel executions determined for that component; and

dynamically running a dataflow graph on multiple nodes to obtain dynamic levels of parallelism by:

for each component,

dynamically assigning a generated instance to one or more nodes of the host systems; and

causing each of the plurality of assigned instances to execute on the one or more nodes of the host systems.

17. The system of claim 10, wherein the communication between the data processing system and the one or more of the host systems executing that configured instance includes:

transmitting, by the given module, monitor data to the second application, and passing, by the second application, the monitor data to the data processing system, wherein the monitor data is configured to be used by the data processing system to track when the given module has completed execution for the data processing system to instruct the another module to execute.

18. The system of claim 10, wherein the given module is a first module, wherein the generated plurality of instances is a plurality of first instances, and wherein the operations further include:

determining, by the data processing system, a number of parallel executions of a second module of the first application;

for the second module, generating a plurality of second instances of the container image in accordance with the number of parallel executions determined for the second module, with each second instance including the first and second applications;

for each second instance, configuring that second instance to execute the second module of the first application;

causing each of the plurality of configured, second instances to execute on one or more of the host systems; and

retrieving, from a local data store, an address of one of the second instances; and

in accordance with the retrieved address, providing, from one of the first instances to the one of the second instances, output from the one of the first instance.

19. One or more machine-readable hardware storage devices for enabling an application in a container image to execute across host systems by assigning instances of the container image to the host system and enabling a portion of the application in an instance on a host system to output data to another portion of the application in another instance on another host system, the one or more machine-readable hardware storage devices storing instructions that are executable by a data processing system to perform operations including:

accessing the container image that includes the first application and a second application, wherein the second application causes a communication between the data processing system and a host system for executing an instance of the container image;

determining, by the data processing system, the number of parallel executions of the first application;

generating a plurality of instances of the container image in accordance with the number of parallel executions determined, with an instance including the first and second applications;

configuring a first instance to execute a first portion of the first application on a first host system;

causing each of the plurality of configured instances to execute on one or more of the host systems;

receiving, from the second application of configured second instance on a second host system, a communication;

based on the communication received from the second application of the second instance on the second host system, transmitting, by the data processing system to the first instance on the first host system, information for the first instance to locate a second portion of the application of the second instance on the second host system to receive an output from the first portion; and

based on the transmitted information to locate the second portion, causing the first portion of the application of the first instance on the first host system to transmit its output data to the second portion of the application of the second instance on the second host system.

20. The one or more machine-readable hardware storage devices of claim 19, wherein the operations further include accessing a specification that specifies a number of parallel executions of a given module.

21. The one or more machine-readable hardware storage devices of claim 19, wherein the container image is a first container image, and wherein the operations further include:

accessing from a hardware storage device an instance of a second container image, wherein the second container image includes the first application and specifies an amount of parallelism for each module in the second application; and

storing the second container image on an execution system.

22. The one or more machine-readable hardware storage devices of claim 21, wherein the operations further include:

transmitting, from the instance of the second container image, a request to an interface included in the execution system to generate the plurality of instances of the container image in accordance with the number of parallel executions determined.

23. The one or more machine-readable hardware storage devices of claim 19, wherein the operations further include:

assigning each of the plurality of configured instances to one or more of the host systems.

24. The one or more machine-readable hardware storage devices of claim 23, wherein assigning is dynamically performed by each configured instance being assigned at run-time and the assignments not being pre-determined.

25. The one or more machine-readable hardware storage devices of claim 24, wherein the first application is a dataflow graph with a plurality of components, wherein the operations further include:

for each component, generating a plurality of instances of the container image in accordance with a number of parallel executions determined for that component; and

dynamically running a dataflow graph on multiple nodes to obtain dynamic levels of parallelism by:

for each component,

dynamically assigning a generated instance to one or more nodes of the host systems; and

causing each of the plurality of assigned instances to execute on the one or more nodes of the host systems.

26. The one or more machine-readable hardware storage devices of claim 19, wherein the given module is a first module, wherein the generated plurality of instances is a plurality of first instances, and wherein the operations further include:

determining, by the data processing system, a number of parallel executions of a second module of the first application;

for the second module, generating a plurality of second instances of the container image in accordance with the number of parallel executions determined for the second module, with each second instance including the first and second applications;

for each second instance, configuring that second instance to execute the second module of the first application;

causing each of the plurality of configured, second instances to execute on one or more of the host systems; and

causing establishment of a communication channel between one of the first instances and one of the second instances, wherein the one of the first instances outputs data and transmits that output data, over the communication channel to the one of the second instances.