syntax = "proto3";

package tensorflow;

import "tensorflow/core/framework/attr\_value.proto";

import "tensorflow/core/framework/full\_type.proto";

option cc\_enable\_arenas = true;

option java\_outer\_classname = "NodeProto";

option java\_multiple\_files = true;

option java\_package = "org.tensorflow.framework";

option go\_package = "github.com/tensorflow/tensorflow/tensorflow/go/core/framework/node\_def\_go\_proto";

message NodeDef {

// The name given to this operator. Used for naming inputs,

// logging, visualization, etc. Unique within a single GraphDef.

// Must match the regexp "[A-Za-z0-9.][A-Za-z0-9\_>./]\*".

string name = 1;

// The operation name. There may be custom parameters in attrs.

// Op names starting with an underscore are reserved for internal use.

string op = 2;

// Each input is "node:src\_output" with "node" being a string name and

// "src\_output" indicating which output tensor to use from "node". If

// "src\_output" is 0 the ":0" suffix can be omitted. Regular inputs

// may optionally be followed by control inputs that have the format

// "^node".

repeated string input = 3;

// A (possibly partial) specification for the device on which this

// node should be placed.

// The expected syntax for this string is as follows:

//

// DEVICE\_SPEC ::= PARTIAL\_SPEC

//

// PARTIAL\_SPEC ::= ("/" CONSTRAINT) \*

// CONSTRAINT ::= ("job:" JOB\_NAME)

// | ("replica:" [1-9][0-9]\*)

// | ("task:" [1-9][0-9]\*)

// | ("device:" [A-Za-z]\* ":" ([1-9][0-9]\* | "\*") )

//

// Valid values for this string include:

// \* "/job:worker/replica:0/task:1/device:GPU:3" (full specification)

// \* "/job:worker/device:GPU:3" (partial specification)

// \* "" (no specification)

//

// If the constraints do not resolve to a single device (or if this

// field is empty or not present), the runtime will attempt to

// choose a device automatically.

string device = 4;

// Operation-specific graph-construction-time configuration.

// Note that this should include all attrs defined in the

// corresponding OpDef, including those with a value matching

// the default -- this allows the default to change and makes

// NodeDefs easier to interpret on their own. However, if

// an attr with a default is not specified in this list, the

// default will be used.

// The "names" (keys) must match the regexp "[a-z][a-z0-9\_]+" (and

// one of the names from the corresponding OpDef's attr field).

// The values must have a type matching the corresponding OpDef

// attr's type field.

// TODO(josh11b): Add some examples here showing best practices.

map<string, AttrValue> attr = 5;

message ExperimentalDebugInfo {

// Opaque string inserted into error messages created by the runtime.

//

// This is intended to store the list of names of the nodes from the

// original graph that this node was derived. For example if this node, say

// C, was result of a fusion of 2 nodes A and B, then 'original\_node' would

// be {A, B}. This information can be used to map errors originating at the

// current node to some top level source code.

repeated string original\_node\_names = 1;

// This is intended to store the list of names of the functions from the

// original graph that this node was derived. For example if this node, say

// C, was result of a fusion of node A in function FA and node B in function

// FB, then `original\_funcs` would be {FA, FB}. If the node is in the top

// level graph, the `original\_func` is empty. This information, with the

// `original\_node\_names` can be used to map errors originating at the

// current ndoe to some top level source code.

repeated string original\_func\_names = 2;

}

// This stores debug information associated with the node.

ExperimentalDebugInfo experimental\_debug\_info = 6;

// The complete type of this node. Experimental and subject to change.

// Currently, the field only contains the return types of the node. That will

// extend in the future to contain the entire signature of the node, as a

// function type.

FullTypeDef experimental\_type = 7;

}