From User's View

Tensor a multi-dimension float-value array. For a Tensor with shape $< a, b, c >$, where a, b, and c are all positive integers, there are in total of abc number of elements, called `size`, in it. The length of the shape is called `rank`. It is also the number of dimensions. Scalar is viewed as a Tensor with shape $< 1 >$, i.e., rank 1.

This leads to the following definition:

```
typedef struct {
  mlvm_uint_t  rank;   /* Must be positive (non-zero). */
  mlvm_uint_t* shape;  /* Length is ‘rank‘ above. */
  mlvm_size_t  size;   /* Total number of elements. */
  double*      value;  /* The value buffer. */
} tensor_t;
```

From Intermediate Representation's View

There are two new concepts.

value mode The first one is ownership, called `value-mode_`, which is an internal field. It represents whether the current Tensor owns its value buffer.

stride The second one is `stride`, with type `mlvm_size_t*`. It is an array with same length as `shape`. For `stride[i]`, it represents the offset difference between the dimension increment at dimension `shape[i]`.

- **tranpose** For a Tensor with shape $< a, b, c >$ and stride $\backslash\{bc, c, 1]$, *tranpose(1, 2, 0)* produces a new Tensor with shape $< b, c, a >$. With `stride` property, we could re-use the same value buffer, saving the copy cost, and represent the new Tensor as: shape $< b, c, a >$ and stride $\backslash\{c, 1, bc]$.

- **broadcast** For a Tensor with shape $< a, 1 >$ and stride $\backslash\{1, 1\backslash\}$, *broadcast*ing to $< a, b >$ is cost free as we just need a new Tensor with shape $< a, b >$ and stride $\backslash\{1, 0\backslash\}$. Similary, *broadcast*ing a Tensor with shape $< 1, b >$ to new shape $< a, b >$ can be represented as shape $< a, b >$ and stride $\backslash\{0, 1\backslash\}$.

- **constant** If all above make sense, a `zeros` tensor with shape $< a, b >$ is simply *broadcast*ing a scalar 0 to $< a, b >$, i.e., stride $\backslash\{0, 0\backslash\}$. So is a `ones` tensor.

- **data access** This would be a big *disadvantage.* Locality is bad for arbitrary stride. Iterating the data value is also quite complicated.

- **reshape** This would be another *disadvantage. Reshap*ing a Tensor with arbitrary stride is not always cost-free. For example, *reshap*ing a Tensor with shape $< a, b >$ and stride $\backslash\{1, 0\backslash\}$ to $< ab >$ needs a value buffer copy.