The paper "DynamicViT: Efficient Vision Transformers with Dynamic Token Sparsification" proposes an attention masking strategy to implement differentiable token pruning in vision transformers. Here's a mathematical explanation of their attention masking implementation:

1. Standard Attention Calculation:
The standard attention matrix A is computed as:
A = Softmax(QK @/" C)

2. Attention Masking Implementation:
To prune tokens while maintaining differentiability, they:

a) First compute the unnormalized attention scores:
P = QK @/" C " ! ^{N×N}

b) Create a binary mask G based on the pruning decisions ˆD:
G_{ij} = {
  1, if i = j (self-connection)
  ˆD_j, if i "` j
}

This means:
- A token always attends to itself (self-loop)
- A token only attends to other tokens if they're not pruned (ˆD_j = 1)

c) Compute the masked attention matrix:
Ã_{ij} = exp(P_{ij})G_{ij} / " _{k=1}^N exp(P_{ik})G_{ik}

Key Properties:
1. Differentiable: The pruning decision ˆD is incorporated through multiplication, maintaining differentiability
2. Hardware Friendly: Maintains fixed tensor shapes (N×N) during training
3. Exact Pruning: When ˆD_j=0, token j has no effect on other tokens (except itself)
4. Self-Loops: Preserve gradient flow to pruned tokens

This implementation allows end-to-end training of both the transformer and the pruning prediction module while achieving actual speedups during inference by completely removing pruned tokens.