```
/home/marco/PycharmProjects/Complex/.venv/bin/python
/home/marco/PycharmProjects/Complex/Tests/Case_ToT.py
All modules loaded.
llama_model_loader: loaded meta data with 23 key-value pairs and 201 tensors from ../Models/tinyllama-
1.1b-chat-v1.0.Q4_K_M.gguf (version GGUF V3 (latest))
llama_model_loader: Dumping metadata keys/values. Note: KV overrides do not apply in this output.
llama_model_loader: - kv   0:                       general.architecture str              = llama
llama_model_loader: - kv   1:                               general.name str              =
tinyllama_tinyllama-1.1b-chat-v1.0
llama_model_loader: - kv   2:                      llama.context_length u32              = 2048
llama_model_loader: - kv   3:                    llama.embedding_length u32              = 2048
llama_model_loader: - kv   4:                         llama.block_count u32              = 22
llama_model_loader: - kv   5:                 llama.feed_forward_length u32              = 5632
llama_model_loader: - kv   6:                llama.rope.dimension_count u32              = 64
llama_model_loader: - kv   7:                llama.attention.head_count u32              = 32
llama_model_loader: - kv   8:             llama.attention.head_count_kv u32              = 4
llama_model_loader: - kv   9:    llama.attention.layer_norm_rms_epsilon f32              = 0.000010
llama_model_loader: - kv  10:                      llama.rope.freq_base f32              = 10000.000000
llama_model_loader: - kv  11:                         general.file_type u32              = 15
llama_model_loader: - kv  12:                       tokenizer.ggml.model str              = llama
llama_model_loader: - kv  13:                      tokenizer.ggml.tokens arr[str,32000]   = ["<unk>", "
<s>", "</s>", "<0x00>", "<...
llama_model_loader: - kv  14:                      tokenizer.ggml.scores arr[f32,32000]   = [0.000000,
0.000000, 0.000000, 0.0000...
llama_model_loader: - kv  15:                  tokenizer.ggml.token_type arr[i32,32000]   = [2, 3, 3, 6,
6, 6, 6, 6, 6, 6, 6, 6, ...
llama_model_loader: - kv  16:                      tokenizer.ggml.merges arr[str,61249]   = ["▁ t", "e
r", "i n", "▁ a", "e n...
llama_model_loader: - kv  17:                tokenizer.ggml.bos_token_id u32              = 1
llama_model_loader: - kv  18:                tokenizer.ggml.eos_token_id u32              = 2
llama_model_loader: - kv  19:            tokenizer.ggml.unknown_token_id u32              = 0
llama_model_loader: - kv  20:            tokenizer.ggml.padding_token_id u32              = 2
llama_model_loader: - kv  21:                    tokenizer.chat_template str              = {% for
message in messages %}\n{% if m...
llama_model_loader: - kv  22:               general.quantization_version u32              = 2
llama_model_loader: - type  f32:   45 tensors
llama_model_loader: - type q4_K:  135 tensors
llama_model_loader: - type q6_K:   21 tensors
print_info: file format = GGUF V3 (latest)
print_info: file type   = Q4_K - Medium
print_info: file size   = 636.18 MiB (4.85 BPW)
init_tokenizer: initializing tokenizer for type 1
load: control token:      2 '</s>' is not marked as EOG
load: control token:      1 '<s>' is not marked as EOG
load: special_eos_id is not in special_eog_ids - the tokenizer config may be incorrect
load: special tokens cache size = 3
load: token to piece cache size = 0.1684 MB
print_info: arch             = llama
print_info: vocab_only       = 0
print_info: n_ctx_train      = 2048
print_info: n_embd           = 2048
print_info: n_layer          = 22
print_info: n_head           = 32
print_info: n_head_kv        = 4
print_info: n_rot            = 64
print_info: n_swa            = 0
print_info: n_embd_head_k    = 64
print_info: n_embd_head_v    = 64
print_info: n_gqa            = 8
print_info: n_embd_k_gqa     = 256
print_info: n_embd_v_gqa     = 256
print_info: f_norm_eps       = 0.0e+00
print_info: f_norm_rms_eps   = 1.0e-05
print_info: f_clamp_kqv      = 0.0e+00
print_info: f_max_alibi_bias = 0.0e+00
print_info: f_logit_scale    = 0.0e+00
print_info: n_ff             = 5632
print_info: n_expert         = 0
print_info: n_expert_used    = 0
print_info: causal attn      = 1
print_info: pooling type     = 0
print_info: rope type        = 0
```

```
print_info: rope scaling     = linear
print_info: freq_base_train  = 10000.0
print_info: freq_scale_train = 1
print_info: n_ctx_orig_yarn  = 2048
print_info: rope_finetuned   = unknown
print_info: ssm_d_conv       = 0
print_info: ssm_d_inner      = 0
print_info: ssm_d_state      = 0
print_info: ssm_dt_rank      = 0
print_info: ssm_dt_b_c_rms   = 0
print_info: model type       = 1B
print_info: model params     = 1.10 B
print_info: general.name     = tinyllama_tinyllama-1.1b-chat-v1.0
print_info: vocab type       = SPM
print_info: n_vocab          = 32000
print_info: n_merges         = 0
print_info: BOS token        = 1 '<s>'
print_info: EOS token        = 2 '</s>'
print_info: UNK token        = 0 '<unk>'
print_info: PAD token        = 2 '</s>'
print_info: LF token         = 13 '<0x0A>'
print_info: EOG token        = 2 '</s>'
print_info: max token length = 48
load_tensors: layer   0 assigned to device CPU
load_tensors: layer   1 assigned to device CPU
load_tensors: layer   2 assigned to device CPU
load_tensors: layer   3 assigned to device CPU
load_tensors: layer   4 assigned to device CPU
load_tensors: layer   5 assigned to device CPU
load_tensors: layer   6 assigned to device CPU
load_tensors: layer   7 assigned to device CPU
load_tensors: layer   8 assigned to device CPU
load_tensors: layer   9 assigned to device CPU
load_tensors: layer  10 assigned to device CPU
load_tensors: layer  11 assigned to device CPU
load_tensors: layer  12 assigned to device CPU
load_tensors: layer  13 assigned to device CPU
load_tensors: layer  14 assigned to device CPU
load_tensors: layer  15 assigned to device CPU
load_tensors: layer  16 assigned to device CPU
load_tensors: layer  17 assigned to device CPU
load_tensors: layer  18 assigned to device CPU
load_tensors: layer  19 assigned to device CPU
load_tensors: layer  20 assigned to device CPU
load_tensors: layer  21 assigned to device CPU
load_tensors: layer  22 assigned to device CPU
load_tensors: tensor 'token_embd.weight' (q4_K) (and 200 others) cannot be used with preferred buffer
type CPU_AARCH64, using CPU instead
load_tensors:   CPU_Mapped model buffer size =   636.18 MiB
llama_init_from_model: n_seq_max     = 1
llama_init_from_model: n_ctx         = 2048
llama_init_from_model: n_ctx_per_seq = 2048
llama_init_from_model: n_batch       = 512
llama_init_from_model: n_ubatch      = 512
llama_init_from_model: flash_attn    = 0
llama_init_from_model: freq_base     = 10000.0
llama_init_from_model: freq_scale    = 1
llama_kv_cache_init: kv_size = 2048, offload = 1, type_k = 'f16', type_v = 'f16', n_layer = 22,
can_shift = 1
llama_kv_cache_init: layer 0: n_embd_k_gqa = 256, n_embd_v_gqa = 256
llama_kv_cache_init: layer 1: n_embd_k_gqa = 256, n_embd_v_gqa = 256
llama_kv_cache_init: layer 2: n_embd_k_gqa = 256, n_embd_v_gqa = 256
llama_kv_cache_init: layer 3: n_embd_k_gqa = 256, n_embd_v_gqa = 256
llama_kv_cache_init: layer 4: n_embd_k_gqa = 256, n_embd_v_gqa = 256
llama_kv_cache_init: layer 5: n_embd_k_gqa = 256, n_embd_v_gqa = 256
llama_kv_cache_init: layer 6: n_embd_k_gqa = 256, n_embd_v_gqa = 256
llama_kv_cache_init: layer 7: n_embd_k_gqa = 256, n_embd_v_gqa = 256
llama_kv_cache_init: layer 8: n_embd_k_gqa = 256, n_embd_v_gqa = 256
llama_kv_cache_init: layer 9: n_embd_k_gqa = 256, n_embd_v_gqa = 256
llama_kv_cache_init: layer 10: n_embd_k_gqa = 256, n_embd_v_gqa = 256
llama_kv_cache_init: layer 11: n_embd_k_gqa = 256, n_embd_v_gqa = 256
llama_kv_cache_init: layer 12: n_embd_k_gqa = 256, n_embd_v_gqa = 256
llama_kv_cache_init: layer 13: n_embd_k_gqa = 256, n_embd_v_gqa = 256
```

```
llama_kv_cache_init: layer 14: n_embd_k_gqa = 256, n_embd_v_gqa = 256
llama_kv_cache_init: layer 15: n_embd_k_gqa = 256, n_embd_v_gqa = 256
llama_kv_cache_init: layer 16: n_embd_k_gqa = 256, n_embd_v_gqa = 256
llama_kv_cache_init: layer 17: n_embd_k_gqa = 256, n_embd_v_gqa = 256
llama_kv_cache_init: layer 18: n_embd_k_gqa = 256, n_embd_v_gqa = 256
llama_kv_cache_init: layer 19: n_embd_k_gqa = 256, n_embd_v_gqa = 256
llama_kv_cache_init: layer 20: n_embd_k_gqa = 256, n_embd_v_gqa = 256
llama_kv_cache_init: layer 21: n_embd_k_gqa = 256, n_embd_v_gqa = 256
llama_kv_cache_init:        CPU KV buffer size =    44.00 MiB
llama_init_from_model: KV self size  =   44.00 MiB, K (f16):   22.00 MiB, V (f16):   22.00 MiB
llama_init_from_model:        CPU  output buffer size =     0.12 MiB
llama_init_from_model:        CPU compute buffer size =   148.01 MiB
llama_init_from_model: graph nodes  = 710
llama_init_from_model: graph splits = 1
CPU : SSE3 = 1 | SSSE3 = 1 | AVX = 1 | AVX2 = 1 | F16C = 1 | FMA = 1 | LLAMAFILE = 1 | OPENMP = 1 |
AARCH64_REPACK = 1 |
Model metadata: {'tokenizer.chat_template': "{% for message in messages %}\n{% if message['role'] ==
'user' %}\n{{ '<|user|>\n' + message['content'] + eos_token }}\n{% elif message['role'] == 'system'
%}\n{{ '<|system|>\n' + message['content'] + eos_token }}\n{% elif message['role'] == 'assistant' %}\n{{
'<|assistant|>\n'  + message['content'] + eos_token }}\n{% endif %}\n{% if loop.last and
add_generation_prompt %}\n{{ '<|assistant|>' }}\n{% endif %}\n{% endfor %}",
'tokenizer.ggml.padding_token_id': '2', 'tokenizer.ggml.unknown_token_id': '0',
'tokenizer.ggml.eos_token_id': '2', 'general.architecture': 'llama', 'llama.rope.freq_base':
'10000.000000', 'llama.context_length': '2048', 'general.name': 'tinyllama_tinyllama-1.1b-chat-v1.0',
'llama.embedding_length': '2048', 'llama.feed_forward_length': '5632',
'llama.attention.layer_norm_rms_epsilon': '0.000010', 'llama.rope.dimension_count': '64',
'tokenizer.ggml.bos_token_id': '1', 'llama.attention.head_count': '32', 'llama.block_count': '22',
'llama.attention.head_count_kv': '4', 'general.quantization_version': '2', 'tokenizer.ggml.model':
'llama', 'general.file_type': '15'}
Available chat formats from metadata: chat_template.default
Using gguf chat template: {% for message in messages %}
{% if message['role'] == 'user' %}
{{ '<|user|>
' + message['content'] + eos_token }}
{% elif message['role'] == 'system' %}
{{ '<|system|>
' + message['content'] + eos_token }}
{% elif message['role'] == 'assistant' %}
{{ '<|assistant|>
'  + message['content'] + eos_token }}
{% endif %}
{% if loop.last and add_generation_prompt %}
{{ '<|assistant|>' }}
{% endif %}
{% endfor %}
Using chat eos_token: </s>
Using chat bos_token: <s>
Iteration 1:
llama_perf_context_print:        load time =     500.04 ms
llama_perf_context_print: prompt eval time =     499.92 ms /    68 tokens (    7.35 ms per token,
136.02 tokens per second)
llama_perf_context_print:        eval time =    2184.62 ms /    99 runs   (   22.07 ms per token,
45.32 tokens per second)
llama_perf_context_print:       total time =    2726.79 ms /   167 tokens
Llama.generate: 28 prefix-match hit, remaining 35 prompt tokens to eval
Explored Thought: 1. Create a cost-saving plan:
Explored Thought: - Identify areas of your business that are costing you more than they should
Iteration 2:
llama_perf_context_print:        load time =     500.04 ms
llama_perf_context_print: prompt eval time =     249.85 ms /    35 tokens (    7.14 ms per token,
140.09 tokens per second)
llama_perf_context_print:        eval time =    2253.72 ms /    99 runs   (   22.76 ms per token,
43.93 tokens per second)
llama_perf_context_print:       total time =    2546.96 ms /   134 tokens
Llama.generate: 27 prefix-match hit, remaining 41 prompt tokens to eval
llama_perf_context_print:        load time =     500.04 ms
llama_perf_context_print: prompt eval time =     308.28 ms /    41 tokens (    7.52 ms per token,
133.00 tokens per second)
llama_perf_context_print:        eval time =    1922.61 ms /    90 runs   (   21.36 ms per token,
46.81 tokens per second)
llama_perf_context_print:       total time =    2271.09 ms /   131 tokens
Llama.generate: 27 prefix-match hit, remaining 52 prompt tokens to eval
```

```
Explored Thought: 2. Start by identifying areas where you can make small improvements, such as reducing
unnecessary expenses or optimizing your workflow.
Explored Thought:
Explored Thought: 1. Determine the root cause of these higher costs.
Explored Thought: 2. Identify and eliminate the underlying factors that are contributing to these costs.
Iteration 3:
llama_perf_context_print:        load time =     500.04 ms
llama_perf_context_print: prompt eval time =     362.02 ms /    52 tokens (    6.96 ms per token,
143.64 tokens per second)
llama_perf_context_print:        eval time =    2132.89 ms /    99 runs   (   21.54 ms per token,
46.42 tokens per second)
llama_perf_context_print:       total time =    2541.26 ms /   151 tokens
Llama.generate: 27 prefix-match hit, remaining 25 prompt tokens to eval
llama_perf_context_print:        load time =     500.04 ms
llama_perf_context_print: prompt eval time =     168.91 ms /    25 tokens (    6.76 ms per token,
148.01 tokens per second)
llama_perf_context_print:        eval time =    1916.37 ms /    89 runs   (   21.53 ms per token,
46.44 tokens per second)
llama_perf_context_print:       total time =    2126.14 ms /   114 tokens
Llama.generate: 27 prefix-match hit, remaining 39 prompt tokens to eval
llama_perf_context_print:        load time =     500.04 ms
llama_perf_context_print: prompt eval time =     272.30 ms /    39 tokens (    6.98 ms per token,
143.22 tokens per second)
llama_perf_context_print:        eval time =    2181.11 ms /    99 runs   (   22.03 ms per token,
45.39 tokens per second)
llama_perf_context_print:       total time =    2500.31 ms /   138 tokens
Llama.generate: 28 prefix-match hit, remaining 42 prompt tokens to eval
llama_perf_context_print:        load time =     500.04 ms
llama_perf_context_print: prompt eval time =     299.17 ms /    42 tokens (    7.12 ms per token,
140.39 tokens per second)
llama_perf_context_print:        eval time =    2124.07 ms /    99 runs   (   21.46 ms per token,
46.61 tokens per second)
llama_perf_context_print:       total time =    2470.64 ms /   141 tokens
Explored Thought: 1. Now that you have identified areas where you can make small improvements, take
action. For example, consider reducing your cable subscription by one package to save money or
automating your recurring billing processes to save time.
Explored Thought:
Explored Thought: 1. "But wait, there's more!"
Explored Thought: 2. "This is just the beginning!"
Explored Thought: 2. Reevaluate the current pricing strategy and consider adjusting it to reflect these
higher costs.
Explored Thought:
Explored Thought: 1. Consider implementing a system of incentives to encourage employees to take more
responsibility for their own health and wellbeing. This could involve offering flexible work
arrangements, such as flexible start and finish times, or paying employees for taking time off work to
exercise, meditate, or engage in other health-promoting activities.
Explored Thought:
================================================================================================
Final Tree of Thoughts:
- Think of a solution to reduce the operational costs of your business.
  - 1. Create a cost-saving plan:
    - 2. Start by identifying areas where you can make small improvements, such as reducing unnecessary
expenses or optimizing your workflow.
      - 1. Now that you have identified areas where you can make small improvements, take action. For
example, consider reducing your cable subscription by one package to save money or automating your
recurring billing processes to save time.
      -
    -
      - 1. "But wait, there's more!"
      - 2. "This is just the beginning!"
  - - Identify areas of your business that are costing you more than they should
    - 1. Determine the root cause of these higher costs.
      - 2. Reevaluate the current pricing strategy and consider adjusting it to reflect these higher
costs.
      -
    - 2. Identify and eliminate the underlying factors that are contributing to these costs.
      - 1. Consider implementing a system of incentives to encourage employees to take more
responsibility for their own health and wellbeing. This could involve offering flexible work
arrangements, such as flexible start and finish times, or paying employees for taking time off work to
exercise, meditate, or engage in other health-promoting activities.
      -
Cost of time:  17.272932767868042
```

```
Process finished with exit code 0
```