青少年组第一轮试题解析

本解析包含 20 道单项选择题和 3 道材料题，共 100 分。每道题后附参考答案与解析。

一、单项选择题（共 20 题，每题 3 分，共 60 分）

第 1 题｜学习范式

某小组请教师为一批作文标注“切题”或“偏题”，再用作文和对应标签训练分类模型。按训练时监督信号的来源，这一过程属于：

A. 无监督学习，因为输入数据是自然语言文本。

B. 监督学习，目标标签来自人工标注。

C. 自监督学习，因为模型会自动计算训练损失。

D. 强化学习，因为每轮训练后都会报告准确率。

答案： B

解析： 训练样本带有人工给出的目标标签，属于监督学习。自动计算损失、输入是文本、报告准确率，都不能据此把训练归为自监督、无监督或强化学习。

第 2 题｜数据划分与评测独立性

数据集中，每名学生提交了多篇作文，其中部分作文是同一篇文章的不同修改版本。现在希望评估模型对训练集中未出现过的学生所写作文的分类效果。以下哪种数据集划分方式最合理？

A. 按作文随机划分，只要每篇作文的文件名不同，就可以视为独立样本。

B. 把每篇作文的初稿放训练集、修改稿放测试集，以保证主题分布一致。

C. 按学生编号分组，同一学生的所有作文和修改稿只进入同一个数据子集。

D. 让每名学生都有作文进入训练集和测试集，以保证测试集覆盖所有学生。

答案： C

解析： 目标是泛化到未见过的学生，因此需要在学生层面隔离数据。按文章或版本划分可能让同一人的写作特征甚至近重复内容跨集合出现，无法直接回答该评估问题。

第 3 题 | 泛化：识别过拟合

在固定的数据划分下继续训练时，某模型的训练损失持续下降，验证损失在达到最低点后持续上升。训练和验证的损失定义相同。对此最合理的判断是：

A. 模型明显欠拟合，因为训练损失还没有停止下降。

B. 验证损失上升说明模型对新样本的泛化能力正在改善。

C. 只要训练损失下降，就应选择最后一个训练轮次的模型。

D. 模型可能开始过拟合，应结合验证表现考虑早停或调整正则化。

答案： D

解析： 训练拟合继续改善而验证表现恶化，是过拟合的典型信号。模型选择应参考独立验证表现，不能只依据训练损失下降，也不能将损失上升解释为性能改善。

第 4 题 | 语言模型目标：下一个词元对齐

用自回归目标训练序列 `[BOS, 我, 喜欢, 音乐, EOS]`。约定每个输入位置预测其紧随的下一个词元；当输入为 `[BOS, 我, 喜欢, 音乐]` 时，对应目标应为：

A. `[EOS, EOS, EOS, EOS]`

B. `[BOS, 我, 喜欢, 音乐]`

C. `[我, 喜欢, 音乐, EOS]`

D. `[音乐, 喜欢, 我, BOS]`

答案： C

解析： 下一个词元目标相对于输入向左对齐一位：BOS 预测“我”，“我”预测“喜欢”，依次直到“音乐”预测 EOS。原样复制、反序或全部预测结束符都不符合题设。

第 5 题 | 语言模型目标：构造掩码样本

某课堂实验规定：每次选中一个词元，用 MASK 替换该位置，并只计算这一位置恢复原词元的损失。以下样本构造符合规定的是：

A. 先保存被选中的原词元作为标签，再把输入中的对应位置替换为 MASK。

B. 输入保留原词元不变，标签也使用原词元，以便模型直接读到目标。

C. 将输入位置和标签都替换为 MASK，让模型学习在该位置输出 MASK。

D. 先遮盖原词元，再随机抽取另一个词元作为恢复目标。

答案： A

解析： 标签需保留原内容，输入中的选定位置则按规则遮盖。把标签也改成 MASK 改变了任务；保留输入原词元违背本实验规定；随机标签不再是恢复原文。

第 6 题｜注意力：权重与内容的区别

只考察 $\mathrm{softmax}(QK^T/\sqrt{d_k})V$ 这一注意力算子。保持 Q、K、缩放因子和掩码不变，只修改 V。下列判断正确的是：

A. Q、K 不变就足以保证输出不变，V 不影响结果。

B. 注意力权重保持不变，但加权求和得到的输出可能改变。

C. V 一改变，原有的因果掩码也会自动改变。

D. 注意力权重必然随 V 改变，而输出一定保持不变。

答案： B

解析： 在题给算子中，权重只由 Q、K、缩放与掩码决定；V 影响被汇总的内容。某些 V 的变化可能在加权后抵消，所以准确表述是输出“可能”改变。

第 7 题｜softmax：整体平移不变性

忽略浮点舍入影响，对同一行的每个有限分数都加上相同常数，再做 softmax。与加常数前相比，结果会怎样？

A. 分数最大的词元会变成概率最小的词元。

B. 所有位置的概率都会增加相同的数值。

C. 各位置的概率保持不变。

D. 无论原分数如何，都会变成均匀分布。

答案： C

解析： 分子、分母的指数项都会乘以同一个常数因子，因此相互抵消。整体平移不改变 softmax；这与把分数乘以某个系数的作用不同。

第 8 题｜因果掩码：训练时的并行计算

训练时完整文本已经给出，程序希望一次前向计算得到多个位置的下一个词元预测损失。以下做法与自回归目标一致的是：

A. 使用因果掩码并行计算各位置，使每个位置只利用当时允许看到的前文。

B. 去掉因果掩码，让每个位置直接读取未来目标，以提高训练准确率。

C. 把所有位置的目标改成序列末词元，就能保持原自回归目标不变。

D. 训练时也必须等模型采样出一个词元后，才能计算下一个位置的损失。

答案： A

解析： 已知训练输入时，可以通过掩码限制依赖并并行计算各位置。逐词生成的依赖不意味着教师强制训练必须逐词采样；去掉掩码会泄漏未来信息。

第 9 题｜主流架构：编码器—解码器

在一个用于翻译的标准编码器—解码器模型中，编码器先处理完整源句。解码器生成译文时，交叉注意力的查询 Q 与键值 K、V 通常分别来自：

A. Q 来自解码器当前表示，K 和 V 来自编码器对源句的表示。

B. Q、K、V 都来自编码器对源句的表示，不使用解码器当前状态。

C. Q、K、V 都来自解码器当前表示，不读取编码器输出。

D. Q 来自编码器对源句的表示，K 和 V 来自解码器当前表示。

答案： A

解析： 交叉注意力让解码器以自身当前状态查询源句表示，因此 Q 来自解码器，K、V 来自编码器。全部取自解码器或编码器的方案混淆了交叉注意力与自注意力。

第 10 题｜主流架构：前缀解码器掩码

某前缀解码器的输入由“已完整给定的提示前缀”和“待生成的回答”组成。约定前缀内部双向可见，回答按自回归方式生成。符合该约定的注意力模式是：

A. 前缀和回答中的所有位置都可看完整序列，包括未来回答。

B. 前缀位置可看完整正确回答；回答位置只能看前缀的第一个位置。

C. 前缀位置可看整个前缀；回答位置可看整个前缀以及不晚于自身的回答位置。

D. 前缀位置只看未来回答；回答位置只能看严格晚于自身的位置。

答案： C

解析： 前缀已经完整给定，内部可双向建模；回答部分需要保持因果性。前缀位置不能通过读取回答把未来答案间接泄漏回回答位置。

第 11 题｜指令数据构建：任务覆盖

一个校园助手需要处理通知问答、表格信息提取和信息不足时的澄清。现有指令数据几乎全是同一类通知问答的改写。下一步最合理的是：

A. 继续大量改写同一条问答，只要总样本数增长就认为任务覆盖充分。

B. 统一把参考答案改成更长的文本，以文字长度代替任务多样性。

C. 按目标任务补充提取、澄清等样本，核验答案，并去除无意义的近重复。

D. 只保留模型目前容易答对的问答，避免数据中出现需要澄清的情况。

答案： C

解析： 数据应覆盖实际要完成的任务和边界情况，且答案需要可靠。表面改写、答案变长或排除困难情境，不能补齐缺失的任务能力。

第 12 题 | Prompt：结构化输出与缺失信息

为“校园通知整理助手”设置角色和提示，要求从活动通知中提取活动名、日期和地点，输出固定 JSON；通知没有写明的字段要填 null。以下提示最符合要求的是：

A. 输出完整的活动档案，日期和地点缺失时请根据常识补齐，避免空值。

B. 你是校园通知整理助手。仅依据通知提取 name、date、location，只输出这三个字段的 JSON；未明确给出的值填 null。

C. 先输出详细分析，再自由选择字段命名，最后给出一个格式大致正确的 JSON。

D. 概括通知的主要信息，并用自然语言说明你认为活动可能在哪里举行。

答案： B

解析： 正确提示同时限定证据来源、字段、输出格式与缺失信息的处理方式。其余方案引入猜测、额外文本或不固定的字段，不能满足既定接口要求。

第 13 题 | 解码：多样化候选生成

一个写作工具希望同一提示能生成多种候选开头。假设模型、提示和其他条件相同，下面哪种做法直接引入了词元选择的随机性？

A. 每一步固定选择概率最高的词元，概率相同时也使用固定的处理顺序。

B. 在每一步按词元概率分布进行采样，并在独立运行中使用独立的随机抽样。

C. 采用固定模板，把模型生成的第一个开头重复返回给每次请求。

D. 按固定得分规则做束搜索，并始终只返回排名第一的候选。

答案： B

解析： 从概率分布采样直接引入随机性，可以产生不同候选，但并不保证每两次结果都不同。固定规则的贪心、确定性束搜索和固定模板都没有这种随机选择。

第 14 题 | 评测指标：摘要重叠与事实正确性

两个摘要模型的 ROUGE 分数接近，但抽查发现，其中一个经常把“报名截止日期”写成原文没有的日期。下一步最合理的评测补充是：

A. 只比较摘要的句子数量，把篇幅更长的摘要视为事实更完整。

B. 只增加 ROUGE 的小数位数，以此判断哪些日期得到了原文支持。

C. 让模型对自己的日期答案重复表达，再以重复次数代替事实核验。

D. 保留文本重叠评测，同时对照原文检查关键事实是否有依据，并记录事实错误。

答案： D

解析： ROUGE 主要反映与参考文本的重叠，不能直接保证每个事实都有原文依据。关键事实需要对照来源核验；更多小数位、篇幅或重复表述都不是事实证据。

第 15 题｜对齐与偏见：受控公平性评测

一个推荐助手应主要依据学习经历和技能推荐课程。研究者怀疑它会因姓名所暗示的性别而给出不同难度的建议。以下评测最能检验这一问题的是：

A. 先删除所有出现推荐差异的档案，再报告剩余档案中的群体一致性。

B. 只统计所有人的平均满意度，只要平均值提高就不再检查不同群体。

C. 让两组档案的技能、经历和姓名都不同，再把推荐差异全部解释为姓名影响。

D. 构造成对档案，保持学习经历和技能一致，只改变姓名这一待研究属性，比较多组推荐结果。

答案： D

解析： 成对控制其他条件能更直接检验待研究属性对推荐的影响。多个条件同时变化会混杂原因；总体平均和删除差异样本都可能掩盖偏见。

第 16 题｜计算：Batch、epoch 与梯度累积

单卡训练集有 1000 条样本，每个 epoch 恰好遍历一次。micro-batch size 为 64，最后不足 64 条的批次也保留。每处理 3 个 micro-batch 更新一次参数；每个 epoch 结束时，若还有不足 3 批的已累积梯度，也进行一次更新，随后清空累积状态。

共训练 5 个 epoch，不跨 epoch 累积梯度。总共进行多少次参数更新?

A. 30 次。

B. 80 次。

C. 25 次。

D. 27 次。

答案： A

解析： 每个 epoch 有 ceil(1000/64)=16 个 micro-batch，更新 ceil(16/3)=6 次；5 个 epoch 共 30 次。25 次漏掉每轮末尾更新，27 次相当于跨 epoch 合并累积，80 是 micro-batch 总数。

第 17 题｜计算：由统计数量求 P/R/F1

某评测集中实际有 80 条正例。模型共预测 120 条为正例，其中 60 条确实为正例。按

`Precision = TP/(TP+FP)`，`Recall = TP/(TP+FN)`，`F1 = 2×Precision×Recall/(Precision+Recall)`

计算，Precision、Recall、F1 依次是多少?

A. 75%、75%、75%。

B. 75%、50%、60%。

C. 50%、75%、62.5%。

D. 50%、75%、60%。

答案： D

解析： TP=60，FP=120−60=60，FN=80−60=20。因此 P=60/120=50%，R=60/80=75%，F1=2×0.5×0.75/1.25=60%。62.5% 是算术平均。

第 18 题｜计算：NDCG 的理想排序范围

某查询的全部 4 篇候选文档，其真实相关等级为 `[3, 2, 1, 0]`。系统只返回前 3 篇，返回顺序对应的等级为 `[1, 2, 0]`。

规定：`DCG@3 = Σ(i=1..3) (2^rel_i−1)/log₂(i+1)`；计算 `IDCG@3` 时，要从全部 4 篇候选中选出相关等级最高的 3 篇并理想排序；`NDCG@3 = DCG@3/IDCG@3`。

取 `log₂2=1`、`log₂3≈1.585`、`log₂4=2`。NDCG@3 最接近：

A. 0.659。

B. 0.797。

C. 1.000。

D. 0.308。

答案： D

解析： DCG=1+3/1.585≈2.893。理想前三篇是 [3,2,1]，IDCG=7+3/1.585+1/2≈9.393，故 NDCG≈0.308。只对已经返回的三篇重排会忽略漏掉的最高相关文档，得到错误分母。

第 19 题｜计算：注意力概率与张量形状

某单头注意力使用 `O = softmax(QK^T/√d_k)V`，softmax 沿键的位置计算。Q、K、V 的形状分别是 `[2,4]`、`[3,4]`、`[3,5]`，没有掩码。

第一条查询对应的未缩放分数，即 QK^T 的第一行，为 `[0, 2ln2, 2ln4]`。其中 ln 是自然对数，`exp(ln a)=a`。

输出 O 的形状，以及第一条查询对第 3 个键的注意力权重，分别为：

A. `[2,5]`，`16/21`。

B. `[2,5]`，`4/7`。

C. `[2,3]`，`4/7`。

D. `[3,5]`，`4/7`。

答案： B

解析： QK^T 的形状为 [2,3]，再乘 [3,5] 的 V 得到 [2,5]。√d_k=2，缩放后第一行为 [0,ln2,ln4]，指数为 [1,2,4]，第 3 个权重为 4/7；16/21 是漏掉缩放的结果。

第 20 题 | 计算：上下文容量与低比特权重

某系统的输入与预留输出合计不能超过 4096 token。系统提示占 320 token，用户输入占 480 token，每个资料片段固定占 256 token，输出需预留 512 token。没有其他输入开销。

该系统另有一个包含 20 亿参数的模型，其权重按每参数 4 bit 存储。本题假设这些 bit 被紧凑打包，忽略量化元数据、KV cache 和其他开销；规定 8 bit=1 byte，1 GB=10^9 byte。

最多可完整放入多少个资料片段？权重数据的理论大小是多少？

A. 10 个，1 GB。

B. 12 个，1 GB。

C. 10 个，2 GB。

D. 11 个，1 GB。

答案： A

解析： 资料预算为 4096−320−480−512=2784 token，floor(2784/256)=10 个。权重存储为 $2\times10^9\times4/8=10^9$ byte，即 1 GB。这里只计算题设中的紧凑权重数据，不把每个 4 bit 参数当成一个 byte。

二、材料题（共 3 题、10 个小题，共 40 分）

材料题一：指令数据清洗与分批

程序填空，4 小题，每题 4 分，共 16 分。每小题只有一个正确选项。

题目材料

某学习小组收集了一批用于指令微调的数据。每条数据都是一个字典，包含字符串字段 `instruction` 和 `output`，分别表示任务指令和参考回答。请补全下面的数据整理程序。

整理规则如下：

1. 去掉指令和回答两端的空白字符，保留中间的空白以及英文字母大小写。
2. 清理后，指令或回答只要有一个为空字符串，就丢弃该条数据。
3. 仅当清理后的指令和回答都相同时，才视为重复，只保留第一次出现的记录。同一指令可以保留不同回答，同一回答也可以对应不同指令。
4. 按原来的先后顺序，每 `batch_size` 条组成一个批次，最后不足一批的记录仍要保留。

保证 `rows` 为列表，其中每个字典都包含上述两个字符串字段；`batch_size` 为正整数。输入可以为空。`str.strip()` 用于去掉字符串两端的空白字符；列表切片包含起点、不包含终点。

```
def prepare_batches(rows, batch_size):
    cleaned = []
    seen = set()

    for row in rows:
        instruction = ____[1]____
        output = row["output"].strip()

        if ____[2]____:
            continue

        key = ____[3]____
        if key in seen:
            continue
        seen.add(key)
        cleaned.append({
            "instruction": instruction,
            "output": output,
        })

    batches = []
    for start in range(0, len(cleaned), batch_size):
        batches.append(cleaned[____[4]____])
    return batches
```

小题

[1] 为满足字符串清理规则，应填入：

A. `row["instruction"]`

B. `row["instruction"].strip()`

C. `row["output"].strip()`

D. `row["instruction"].strip().lower()`

[2] 为丢弃缺少有效指令或有效回答的数据，应填入：

A. `not instruction and not output`

B. `instruction and output`

C. `instruction == output`

D. `not instruction or not output`

[3] 为按规则识别重复数据，应填入：

A. `(instruction, output)`

B. `instruction`

C. `output`

D. `(len(instruction), len(output))`

[4] 为正确分批且保留最后不足一批的记录，应填入：

A. `0 : batch_size`

B. `start + batch_size : start`

C. `start : start + batch_size`

D. `start : batch_size`

参考答案与解析

答案：**B、D、A、C**。

[1] 答案：B

`strip()` 正好完成两端空白清理。A 没有清理；C 取错字段；D 会改变英文大小写，不符合保留原大小写的要求。

[2] 答案：D

只要两个字段中任意一个为空就丢弃，因此用 `or`。A 仅丢弃两者都为空的记录；B 会跳过有效记录；C 错把指令与回答文字相同当作无效，例如复述任务允许二者相同。

[3] 答案：A

二元组同时保存两个字段，能区分“同指令、不同回答”和“不同指令、同回答”。B、C 分别只看一个字段；D 只比较长度，不同文本也可能长度相同。

[4] 答案：C

每批从当前 `start` 开始，到 `start + batch_size` 之前结束；超出列表长度的切片会自然截到末尾。A 每次重复第一批；B 的起点大于终点，正向切片为空；D 把批次大小误当成固定终点，第二批起就不正确。

考点小结：指令数据质量、字符串、布尔条件、元组去重、列表切片。区分点在于严格遵守清理和去重规则，不依赖模型或框架知识。

材料题二：批量检索与无效文档过滤

程序填空，3 小题，每题 4 分，共 12 分。每小题只有一个正确选项。

题目材料

某 API 文档检索系统一次接收多条查询。系统已得到查询的词元向量和文档向量，但有些文档已经过期，需要在检索时排除。请补全程序，为每条查询返回最相似的 `k` 篇有效文档。

本题使用 PyTorch，所有张量均位于 CPU。输入含义如下：

输入	形状	含义
`query_hidden`	`[B, L, H]`	B 条查询，每条补齐到 L 个词元，每个词元向量 H 维
`query_mask`	`[B, L]`	布尔张量，True 为真实词元，False 为补齐位置
`document_vectors`	`[N, H]`	N 篇文档的向量
`document_valid`	`[N]`	布尔张量，True 为有效文档

检索流程如下：先对每条查询的真实词元做平均池化，再将查询向量和文档向量分别做 L2 归一化。归一化向量的点积就是余弦相似度。过期文档的分数应设为负无穷，最后分别取每条查询分数最高的 `k` 项。

补充说明：`unsqueeze(-1)` 在最后添加长度为 1 的维度，使 `[B,L]` 的 mask 可以与 `[B,L,H]` 逐元素相乘；布尔值参与乘法时 True/False 按 1/0 处理。`sum(dim=1)` 沿词元维求和，`keepdim=True` 保留长度为 1 的维度。`masked_fill(mask, value)` 将 mask 为 True 的位置替换为 value；`~` 对布尔张量取反。`topk` 默认选择最大的 k 项，并返回分数与下标。

保证 B、L、H、N 均为正整数，每条查询至少有一个真实词元；文档向量及正确池化后的查询向量均非零；有效文档数不少于 k，且 `1 <= k <= N`。同一查询对应的有效文档分数互不相等。补齐位置的向量不保证为零，相似度也可能为负数。

```
import torch
import torch.nn.functional as F

def batch_retrieve(query_hidden, query_mask,
                   document_vectors, document_valid, k):
    query_vectors = ____[1]____
    query_vectors = F.normalize(query_vectors, p=2, dim=-1)
    document_vectors = F.normalize(document_vectors, p=2, dim=-1)

    scores = query_vectors @ document_vectors.T
    scores = ____[2]____
    top_scores, top_indices = ____[3]____
    return top_indices, top_scores
```

小题

[1] 为仅对真实词元做平均池化，得到形状为 `[B,H]` 的查询向量，应填入：

A. `query_hidden.mean(dim=1)`

B. `query_hidden.sum(dim=2) / query_mask.sum(dim=1, keepdim=True)`

C. `(query_hidden * query_mask.unsqueeze(-1)).sum(dim=1) / query_mask.sum(dim=1, keepdim=True)`

D. `query_hidden[:, 0, :]`

[2] 为确保所有过期文档均不会进入返回结果，应填入：

A. `scores.masked_fill(~document_valid.unsqueeze(0), float("-inf"))`

B. `scores.masked_fill(document_valid.unsqueeze(0), float("-inf"))`

C. `scores * document_valid.unsqueeze(0)`

D. `scores.masked_fill(~document_valid.unsqueeze(0), float("inf"))`

[3] 为分别返回每条查询的 k 个最相似文档，应填入：

A. `torch.topk(scores, k=k, dim=0)`

B. `torch.topk(scores, k=k, dim=1, largest=False)`

C. `torch.sort(scores, dim=1, descending=True)`

D. `torch.topk(scores, k=k, dim=1)`

参考答案与解析

答案：C、A、D。

[1] 答案：C

mask 扩展后为 `[B,L,1]`，乘法排除补齐位置；沿 L 维求和得到 `[B,H]`，再除以每条查询的真实词元数 `[B,1]`。A 把补齐位置也计入；B 沿 H 维求和得到 `[B,L]`，维度和含义都错误；D 只取首词元，不是题设要求的平均池化。不能因为某些模型使用首词元池化，就替换本题明确规定的池化方式。

[2] 答案：A

无效文档需要被排除，因此对 `~document_valid` 对应的位置填负无穷，形状 `[1,N]` 会广播到所有查询。B 排除了有效文档；C 只是将无效文档置零，当有效文档分数为负时，无效文档仍可能胜出；D 把无效文档变成最高分。

[3] 答案：D

`scores` 为 `[B,N]`，需要沿文档维 N 即 `dim=1` 取最大值。A 沿查询维比较；B 取最不相似文档；C 返回全部 N 项，在 k 小于 N 时不满足输出应为 `[B,k]` 的要求。

考点小结：带 mask 的池化、广播、向量相似度、无效项过滤、按正确维度 top-k。扩展背景为向量检索，所需接口和维度均已给出。

材料题三：Parason——推理分头进行，时间怎样计算？

请阅读下面根据论文 **Parason: Revealing Subtask- and Trial Parallelism in LLM Reasoning**（2026 年 8 月 25 日，arXiv v1）整理的中文阅读材料，然后回答 3 个问题。每小题 4 分，共 12 分；每小题只有一个正确选项。

材料为面向考试的中文译写，包含研究问题、方法、实验数据和结论边界。论文实验与题目中的应用情境分别标明。

中文阅读材料

本文根据 Zhengyang Zhang 等人的论文 Parason: Revealing Subtask- and Trial Parallelism in LLM Reasoning（2026 年 8 月 25 日，arXiv v1）译写。材料综合正文的方法、实验和相关讨论，补充了并行计算术语解释；不是论文全文的逐字翻译。表格采用原论文表 3、表 2 的真实数据，并引用表 5 的实测结果；小题中的分支长度为题目给定。原文采用 CC BY 4.0 许可：论文原文；许可说明。

1. 推理越长，等待就一定越久吗

语言模型生成文本时，通常逐个生成词元。词元是模型处理文本的基本单位，不一定对应一个完整的汉字或单词。模型可能先写出条件分析，再尝试一种解法，遇到困难后换一条路线，最后整理答案。这些中间过程有助于解题，但如果全部依次进行，较长的推理也意味着较长的等待。

一个自然的问题是：这些步骤是否必须排成一队？如果后一步依赖前一步的结果，就必须等待；如果两部分可以独立完成，就可能同时进行。并行计算的关键不是简单增加执行者数量，而是识别哪些工作之间没有必须遵守的先后依赖，再把它们的结果正确合并。

Parason 研究如何让模型把一段推理组织成可以执行的并行结构。它没有假设所有思考都能同时完成，而是保留必要的顺序步骤，在其中找出可以分开的局部区域。这样的研究需要同时关心答案准确率、总生成量和影响等待时间的最长路径。

2. 两种不同的并行关系

第一种是“子任务并行”。模型把问题分解为几个互相独立、但最终都需要完成的部分。例如，一道题分别要求计算两个独立条件下的数值，再把它们代入最终公式；两项计算可以并行，最后却必须拿到两项结果。论文把这种关系称为 AND，即“这些部分都需要”。

第二种是“尝试并行”。模型面对不确定的解题方向，同时探索不同假设或路线。例如，一条路线尝试代数变换，另一条路线寻找几何关系；它们是解决同一困难的替代尝试，不要求每条路线都成功。论文把这种关系称为 OR，即“不同路线可能提供可用解法”。

这里的 AND 与 OR 描述的是分支在推理中的作用，不能直接当作运行系统的等待规则。Parason 当前的合并方式会把尝试分支的输出全部接回后续推理上下文，让模型综合这些探索记录，而不是自动采用第一个成功分支并取消其他分支。失败路线也可能说明某个假设行不通，为后续推理提供信息。

这与把完整问题独立回答多遍也不完全相同。完整重复采样可能反复生成相同的前半段；在一条推理内部并行，可以先共享已经得到的分析，再从需要分岔的位置展开不同任务或尝试。共享前缀、分支执行和结果合并，共同组成了一次完整推理。

3. 怎样把文字安排变成真实执行

如果模型只在回答里写“现在同时计算两件事”，执行系统仍然逐字生成，速度不会因此自动提高。Parason 使用一套明确的结构标记，告诉运行系统哪里开始并行、各分支的目标是什么、分支属于子任务还是替代尝试，以及哪里重新汇合。

例如，<Parallel> 标记并行区域，<Subtask> 和 <Trial> 区分两种分支目标。运行系统需要能解析这些结构，并通过工具调用派发独立工作。工作分支完成后，系统把输出交回模型，再继续顺序推理或产生答案。论文还用语法约束减少结构不合法的输出，避免模型只生成看起来类似标记、却无法执行的文本。

结构可解析并不代表任务分解一定正确。如果第二个分支实际需要第一个分支尚未得到的数值，却被当作独立工作同时启动，它就可能缺少必要信息。反过来，把完全独立的工作仍然串行安排，也会丢失可利用的并行机会。因此，训练不仅要让模型学会标记格式，还要让它学会识别依赖和组织有用的分支。

4. 让模型学习怎样分开推理

研究者以已有的 964 条 Qwen3-8B 推理轨迹为基础，整理并行训练数据，区分子任务和尝试分支，并为尝试阶段补充探索路线。模型先进行监督微调，也就是学习这些结构化示例，掌握怎样输出符合要求的推理过程。随后，在约五万三千道复杂推理问题上进一步进行强化学习。

论文把后一个阶段的方法称为 PA-GRPO。读者无需掌握其优化公式，只需理解奖励考虑的几类因素：最终答案是否正确，最长生成路径是否过长，以及是否产生了有用的子任务和尝试结构。奖励系数决定训练对这些因素分别重视了多少。

子任务分支主要把必要工作从串行队列中移出去；尝试分支则可能通过增加探索宽度提高找到正确解法的机会，同时增加总生成量。鼓励尝试与鼓励缩短路径并不是同一个目标。奖励设计需要平衡这些作用，不能简单地把“分支越多”当作越好，也不能为了更短的过程而放弃回答正确性。

5. 总生成量与最长路径

假设一段推理先有共同前缀，随后进入一个并行区域，最后再汇总结果。总生成量要把前缀、所有分支和汇总部分全部相加，因为这些内容都实际生成了。即使某个尝试分支没有找到解法，它已经生成的内容也不能从总量中抹去。

最长路径则采用另一种计算方法。前缀和汇总必须先后完成，正常相加；一个同时启动、等待全部分支结束的并行区域，贡献的是最长分支的长度，而不是所有分支长度之和。这条从开始到结束、累计长度最大的依赖路径，也称为关键路径。

缩短不在关键路径上的分支，未必能减少整段等待时间。假如其他条件不变，最慢分支仍然占据相同时间，那么一个早已结束的短分支再快一点，也不会使汇合提前。相反，给原本较短的分支增加一些有用检查，只要没有使它成为最长分支，就可能增加总工作量而不增加理想化的关键路径长度。这解释了为什么总计算量和等待时间需要分别报告。

本文的小题采用理想化条件：各分支生成速度相同、可以真正同时运行，且忽略启动、通信和额外处理开销。在这些条件下，最长路径的词元数可以帮助估计相对等待时间；若把完全相同的工作串行执行，总生成量除以最长路径长度，可以得到理想化的加速比。比较时应保持所完成的工作相同，不能把一种方法多探索的分支当作另一种方法本来就必须执行的工作。

6. 有限等待预算下能答对多少

论文在多种数学推理测试上报告最终答案准确率。下面节选自原论文表 3，考查 AIME24。在这个实验里，预算 B 限制的是最长生成路径的词元数，不是所有分支生成词元的总和。SFT 表示只进行监督微调；另外两行在此基础上使用 PA-GRPO，括号中为对子任务并行的奖励系数。

模型配置	B = 2,048	B = 8,192	B = 16,384	B = 24,576
仅 SFT	16.8%	41.8%	66.8%	70.2%
PA-GRPO，子任务系数 0.050	26.8%	60.0%	72.2%	74.8%
PA-GRPO，子任务系数 0.100	34.7%	60.3%	74.4%	75.7%

比较同一列，可以观察在相同最长路径预算下的准确率差异；比较同一行，则可以观察同一配置在不同预算下的变化。两种比较回答的问题不同。如果同时改变模型配置和预算，就需要把两项变化都说明，而不能把结果差异全部算在其中一项上。

表格也允许按任务要求选择配置。例如，如果先规定必须达到某个准确率，再在列出的预算中寻找可用选项，这是一种有约束的比较。但“在表中列出的预算里最小”，不等于研究者已经测试了所有可能的预算，更不等于找到任何情况下的全局最优方案。

7. 准确率与等待时间可能怎样取舍

下表节选自原论文表 2 的三个配置。准确率和最长路径词元数均按 AIME24、AIME25、Math500、AMC 四个测试取平均。k 表示一千；最长路径越短越好，准确率越高越好。两种强化学习配置使用不同的奖励设置，因此这张节选表主要用于比较取舍，不把差异归因于唯一一个系数。

配置	四个测试的平均准确率	平均最长路径词元数
Parason-8B，仅 SFT	82.2%	13.2k
PA-GRPO，原始设置，尝试系数 0.100	84.7%	14.5k
PA-GRPO，延迟惩罚系数 0.1，尝试系数 0.025	83.8%	12.1k

这里，平均准确率最高的配置并不是路径最短的配置。相对于仅 SFT，原始设置提高了准确率，但平均路径也更长；带有上述延迟惩罚设置的配置则在这组数据中同时取得更高准确率和更短路径。选择哪一种，要结合应用对正确率和等待时间的要求，不能仅依据它是否使用了强化学习。

两张表采用的统计范围也不同：AIME24 预算表只涉及 AIME24 且设置了固定预算，四测试平均表现表则是四个测试的平均表现。跨表比较前应先核对数据含义，不能把四测试均值当作 AIME24 预算表某一预算下的准确率。

8. 实际秒数还需要另外测量

最长路径是一种有用的结构指标，但词元数不是秒数。实际系统还会受到设备数量、各分支共享计算资源、生成速度、任务启动和结果合并等因素影响。特别是合并更多分支会带来更多上下文，真实运行中处理这些内容也可能有开销。

计算最长路径时还隐含了资源能够支持相应并发的条件。如果模型安排了多个独立分支，设备却只能一次运行其中一部分，就会产生排队；文本结构仍然是并行的，运行过程却未必完全并行。因此，比较不同系统的实测耗时，应说明设备和执行条件。相同的路径词元数也不保证相同秒数，因为不同运行配置下每个词元的生成速度可能不同。

论文并非只报告词元指标。表 5 还在 A800 GPU 上测量了一次 AIME24 实验的真实运行时间：简单题组的并行与串行时间分别为 188.3 秒和 304.2 秒，相应实测加速约为 1.62 倍，而该组的词元级加速比为 1.70 倍。两种数值并不完全相同，说明从结构上的并行机会到实际速度收益之间仍隔着运行条件。

因此，这项工作展示的是如何识别并执行推理中的并行部分，以及在给定模型和测试条件下得到怎样的准确率与效率取舍。它不能直接保证任意问题都可以平分给多个分支，也不能保证开三个分支就快三倍。理解分支的依赖关系、等待规则和评价单位，才能正确解释论文中的效率数字。

小题

[1] 方案甲同时计算最终公式需要的两个独立中间量；方案乙同时尝试两条可能的证明路线。按照材料对 Parason 的说明，下列哪项正确？

A. 甲属于尝试并行，乙属于子任务并行；二者都只保留最先结束的分支。

B. 甲属于子任务并行，乙属于尝试并行；乙被称为 OR，因此系统一定会在任一分支成功时立刻取消其他分支。

C. 甲属于子任务并行，乙属于尝试并行；当前方法仍把乙的各分支输出合并到后续上下文，OR 不等于“最先成功即停止”。

D. 甲、乙都不能并行，因为任何推理中的两条路线都必定互相依赖。

答案： C

答案解析： 甲的两个中间量都为最终结果所需，对应子任务并行的 AND 关系；乙是相互替代的解题探索，对应尝试并行的 OR 关系。Parason 当前仍收集全部尝试分支，供后续综合，因此不能从 OR 的名称推断存在“首个成功即取消其他分支”的运行机制。

[2] 根据“有限等待预算下能答对多少”一节的 AIME24 预算表，只考虑“仅 SFT”和“PA-GRPO，子任务系数 0.100”，要求 AIME24 准确率至少为 60%，并从表中列出的预算中选最小值。以下哪项正确？

A. 两种配置都应选 8,192，因为这一预算一定能让任何配置达到 60%。

B. 仅 SFT 应选 16,384，PA-GRPO 应选 8,192；后者所需的最小列出预算是前者的一半，但这不能直接证明总生成量或真实耗时也减半。

C. 仅 SFT 应选 24,576，PA-GRPO 应选 16,384；只有准确率严格大于 70% 才符合本题要求。

D. 仅 SFT 应选 16,384，PA-GRPO 应选 8,192；由于预算减半，可以确定设备费用、总生成词元和真实耗时都恰好减半。

答案： B

答案解析： 仅 SFT 在 8,192 预算下为 41.8%，未达到要求；在 16,384 下为 66.8%，是列出预算中第一次达到要求。指定的 PA-GRPO 在 2,048 下为 34.7%，在 8,192 下为 60.3%，所以选 8,192。预算比为 8,192 ÷ 16,384 = 1/2，但这里的预算只限制最长路径，不能直接换算成总生成量、费用或实测秒数。

[3] 一次推理先生成 120 个词元的共同前缀，然后同时启动三条尝试分支，长度分别为 80、140、100 个词元；80 词元分支已经给出可用路线，但系统仍按材料所述规则等待并收集合并全部分支，最后生成 60 个词元的汇总。忽略额外开销，各分支速度相同。本次总生成量、最长路径长度分别是多少？

A. 500、500；只要生成过的词元都必须算入最长路径。

B. 260、260；一条路线可用后，其余分支实际生成的词元都可以不计。

C. 500、260；总量需要计算全部分支，但最长路径应使用最短分支。

D. 500、320；总量相加全部分支，最长路径使用 140 词元的最长分支。

答案： D

答案解析： 总生成量为 120 + 80 + 140 + 100 + 60 = 500。最长路径为 120 + max(80, 140, 100) + 60 = 320。可用路线较早出现，并没有改变题目明确规定的等待与合并规则。

考点小结： 任务依赖；子任务并行与尝试并行；总工作量与关键路径；带约束的表格比较；理论指标和实测性能的区别。无需理解数学竞赛题内容或强化学习算法推导。