两周内完成 13 万行形式化拓扑：人人都能用的简单、低成本自动形式化?

Josef Urban ✉AI4REASON 与哥德堡
大学

摘要

本文简要介绍一个项目，该项目已将 Munkres 教科书中的大部分一般拓扑学内容自动形式化（全书共 7 章、39 节，241 页）。项目自 2025年11月21日 启动，截至 2026年1月4日，已产出 16 万行形式化拓扑内容。其中大部分（约 13 万行）是在两周内完成的，从 12月22日 到 1月4日，大语言模型订阅费用约为 100 美元。这包括乌雷松引理的 3000 行证明、乌雷松度量化定理的 2000 行证明、Tietze 延拓定理的逾一万行证明，以及许多其他成果（总计超过 1500 条引理和定理）。

这种方法相当简单且成本低廉：在大语言模型与配备核心基础库、速度足够快的证明检查器之间，构建一个长期运行的反馈循环。这里使用的大语言模型是通过各自的 Codex 或 Claude Code 命令行界面运行的 ChatGPT（主要是 5.2 版）或 Claude Sonnet（4.5 版）。证明检查器是 Chad Brown 的高阶集合论系统 Megalodon，核心库则是 Brown 对基本集合论和超现实数（包括实数等）的形式化。其余部分是一些提示工程和技术选择，我们将在此介绍。鉴于进展迅速、成本低廉、所用的交互式定理证明系统及其库几乎无人知晓，且人人都能采用这种简单的配置，我们相信，到了 2026 年，（自动）形式化或许会变得相当容易且无处不在，而不论使用哪种证明助手。

2012年 ACM 学科分类 计算理论 → 自动推理；计算理论 → 高阶逻辑；计算理论 → 逻辑与验证

关键词与短语 自动形式化、自动推理、交互式定理证明、形式化证明助手、机器学习、语言模型

数字对象标识符 10.4230/LIPIcs...

资助 本研究得到欧洲研究理事会 NextReason 项目、亚马逊研究奖、捷克教育、青年和体育部 ERC CZ 项目 *POSTMAN* LL1902 号、捷克科学基金会 25-17929X 号资助，以及 AI4REASON研究所赞助方的支持。[1]

arXiv:2601.03298v1 [cs.LO] 6 Jan 2026

1 引言

> “我们当中有些人认为，许多本可以轻易为 *QED* 项目作出贡献的人，却被 *AI* 或 *AR* 的诱人魅力分散了注意力。”
>
> – QED宣言

注意事项:

这是一份关于正在进行的实验的初步说明（主要是为了让其他人也能在自己的环境中尝试）。随着实验推进，我的评估可能会有所变化。

[1] https://ai4reason.eu/sponsors.html

© Josef Urban；依据知识共享署名 4.0 国际许可协议授权莱布尼茨国际信息学会议论文集 达格施图尔城堡——莱布尼茨信息学中心，达格施图尔出版，德国
LIPICS

XX:2 13万行形式拓扑，两周

事情是这样开始的：

自 2014[2] [7, 6]以来，我一直和同事们（或多或少认真地）研究学习辅助的自动形式化。关于自动形式化的历史，我在 2024 年波恩的演讲[3] 及其幻灯片中略有介绍。[4] 长话短说，自 2005 年 MaLARea 的首次实验6 [9],以来，我逐渐深信推理／逻辑／演绎与学习之间反馈回路的力量，也相信我们在形式化数学时，本身就处于这样的反馈回路之中（例如，参见 [10] 中的最后两句话）。2018 年，神经语言模型（尤其是基于注意力机制的模型）在我们首次针对大型合成 Mizar 到 LaTeX 语料库进行的实验中，展现出令人惊讶的非形式化到形式化的转换能力 [12]，这让我进一步刮目相看。我开始相信自己关于自动形式化的宣传，也就是说，自动形式化将远比我在 2014 年保守估计的 25 年时间跨度更早变得实用。[5] 自 2020 年以来，这一领域在拥有更多资源的产业团队中日益流行；看到他们大力投入[6]之后，我（除了一些例外）主要转而关注自动定理证明和猜想生成中其他（资源需求较低的）反馈回路 [11, 5, 3, 4, 8]。ChatGPT（及类似的大语言模型）问世后，我偶尔尝试让它们对篇幅较短的论文进行自动形式化（例如那篇关于 Ramsey(3,6) 的四页论文 [2]），但起初它们似乎不太擅长完成更深入、耗时更长的工作——至少在没有为其搭建一些基础设施的情况下是这样。

启动本文所述实验的直接契机，是 Zar Goertzel 分享了他使用 Codex 和 Claude Code 等编码智能体进行自动形式化并取得良好进展的情况。尤其是在 2025年下半年（与本研究同期），Zar 使用 Claude Code，耗费约 1.3 万行 Lean 代码，完成了我那篇关于 Ramsey(3,6) 的测试论文的完整 Lean 证明。[7] 经历几次尝试未果，并与 Chad Brown 简短讨论后，我决定开展当前这项实验：使用 Megalodon 证明助手和基于大语言模型的命令行编程智能体对拓扑进行形式化。

2 当前设置

最新设置自 12月22日以来基本保持稳定，并用于（几乎）自动化的形式化工作，最终产出了约 13 万行代码，以及上文提到的已完整证明的主要定理。设置如下：

1. 订阅 ChatGPT Pro（每月 200 美元），即可使用 OpenAI 的 Codex 编码智能体。**2.** 我始终使用 ChatGPT 5.2 模型，并将推理设置为高（而非极高）。[8]**3.** 我使用 Linux 的 bubblewrap (bwrap[9]) 沙箱隔离工具隔离编码智能体。其实，我很可能也可以使用其他合理的沙箱隔离方案。我想赋予编码

[2] 我可能是在 CICM’14 https://t.ly/ 演讲 中 PaeKH [7]

[3] https://www.youtube.com/watch?v=4JeezEGc_gQ [4] http://grid01.ciirc.cvut.cz/~mptp/bonn24_af.pdf

[5] http://ai4reason.org/aichallenges.html [6] 有时这说法或许有些牵强——请看 https://archive.ph/8Ib1k 上关于“引入自动形式化”的说法。

[7] https://github.com/zariuq/ai-agents/tree/main/lean-projects/ramsey36/Ramsey36 [8] 资源更多的用户可以尝试用其他设置重复这项实验，并进行更多消融实验。我没有使用额外的高档设置，因为仅高推理模式就会在不到一周内耗尽每周使用限额。

[9] https://github.com/containers/bubblewrap

智能体不受限制的权限（可以调用各种危险工具），因此我将它们隔离在沙箱中。我把外部的几个目录映射到沙箱中，其中包含各种二进制文件和项目数据。到目前为止，智能体在沙箱中还没有造成严重损害，但已经发生过几次状况（例如，智能体弄不清数据位置、文件名等），让我庆幸自己做了隔离。我会定期从外部备份项目数据目录。

4. 在 bwrap 中，我运行命令行编码智能体（现在用的是 Codex，不过我也可以并且确实用类似的方法和结果运行过 Claude Code）。
5. bwrap 本身运行在 tmux 终端管理器／复用器中，通常是在一个长期运行（数天）的会话里，并从外部记录和交互。我很可能也可以使用其他类似工具来完成这项工作，例如 GNU Screen。

6. 我使用 Megalodon（高阶集合论）证明检查器二进制文件来检查证明。所用的二进制文件来自 Megalodon Wiki（mgwiki）的 GitHub 仓库，这确保了与 mgwiki 工具链兼容。这样一来，任何能够编译的形式化内容都可以推送到 mgwiki，并立即转换为 HTML。这种带有交叉链接的 HTML 展示让我们可以轻松研究正在发展的形式化内容。

7. 作为核心库，我使用了 Brown 对基础集合论和超现实数（包括实数等）的形式化。我们删除了其中大部分证明，并用“admit.”替代，这样初始库文件就更小，也（希望）更便于大语言模型研究。不过，我保留了 Brown 的若干证明（例如 Diaconescu 排中律证明），以便大语言模型（希望）能够学习如何在 Megalodon 中证明命题。

8. 就是这样。其余部分是一些提示，以及下文所述的其他工具（可选，后续添加）。

3 当前提示

除少数例外情况外，自12月22日起，每当 Codex 报告已完成工作时（也就是每当 Codex CLI 将控制权交还给用户并请求新的提示时），我都会反复向它提供以下简短的固定提示：

```
Read the file CLAUDE.md .  Treat it as authoritative work instructions.  Follow
those instructions exactly for all subsequent actions and responses.  That means
work as long as possible (as specified) without stopping.
```

据我估计，在这两周内，这条提示已被自动提供给 Codex 约1000至2000次。[10] 目前（第14版）的“工作规则”文件 CLAUDE.md 如下：

```
# Rules for Working on Math_Background.mg

## STRICT PROHIBITIONS

### **ONLY edit in the topology secti
    on**- The topology section starts at **line 6495** (`Section Topology.`)
- Everything after line 6495 is by default in the Topology section and can be
edited (even very high line numbers).- **NEVER edit anything before line 6495**
- This includes set theory, cardinal arithmetic, ordinals, etc.
```

[10]解析 Codex 日志后，应该可以精确计算出这一数字。

XX:4 13万行形式拓扑，两周

```
### Before ANY edit to Math_Background.mg:
1. Verify the line number is >= 6495 (topology section)2. If unsure,
 run: `grep -n "^Section Topology" Math_Background.mg`
3. If the target is before line 6495, **STOP** and ask the user

## When Resuming from Summary
1. **READ THIS FILE FIRST** before making any edits
2. Verify you understand which section you're allowed to edit
3. Double-check line numbers before proceeding

## Never throw away useful work
1. You should almost never revert to previous backups (you can use temporary
admits when something is hard).2. If there is a really major reason for reverting,
 you have to salvage allthe useful work done in between.3. In particular,
 you have to ensure that all compiling theorems and
definitions added since are kept.4. If you later discover such screwup,
 you have to immedially start workingon salvaging the lost work.
5. This rule exist because you have acidentally thrown away 9k lines (more
than a day) of work between bck2658 and bck2659 (by mysteriously justifying
it as "reverting to colleague baseline"). This must never ever happen again.
6. **SIMPLE CHECK**: after each backup, run wc on the current and previous backup.
If the current backup is smaller, you have to explicitly justify (in the
CHANGES file) the decrease and explain that you have not thrown out useful work.

## Megalodon Language Details

### Logic System
- **This is classical Megalodon** (not constructive)
- Law of excluded middle (xm) is available
- Can use proof by contradiction freely

### Syntax Rules
- **`/\` is left-associative**: `A /\ B /\ C` means `((A /\ B) /\ C)`
- **Only use `(** ... **)` for comments** - no other comment syntax; no `*` inside comments
- When working with nested conjunctions,  remember the left-associative structure

## Work Strategy
- You can do the following things in any order but you should always
progress and produce some more code.
- Keep carefully fixing any incorrect/bad definitions/theorems you
find (note that this may lead to fixing some proofs, etc).- Keep eliminating axioms and stubs,
 replacing them with more complete
theorems and definitions (gradual/partial approaches are ok when needed).
- Keep removing admits in unfinished proofs and producing more complete
proofs. This may also lead to adding more auxiliary lemmas/theorems.- While doing the above,
 remember that:- Doing easy things is initially OK, however, don't be afraid to try
to do (gradually/partially if needed) some harder theorems too. Don't
endlessly jump around looking for easy things - that wastes time.
- Balance simple infrastructure theorems with more challenging results
- Your **strong focus** should be on finishing the major well-known theorems
ASAP (across all sections). Prioritize them (even if they are hard) over doing
many examples/exercises.
```

- Use gradual/partial approaches for difficult theorems when needed (and don't delete such started partial proofs - use temporary "admit" in their various branches and keep gradually eliminating those). Also, structure bigger proofs into useful top-level/helper lemmas wherever possible.
- Frequently lookup the necessary notions defined before the topology section. Those should be completely trusted and built upon. They are listed in TRUSTED_DEFS.txt – Also, completely trust every theorem in the pre-topology section (and never attempt to re-prove it). We do have their proofs but have admited them in this document for brevity – Also, always grep all current Definitions and Theorems in the topology section before creating a new one. Be sure to remove/avoid duplicities.
- Do not introduce axioms that are exactly the same as the theorems you want to prove
- it only duplicates and pollutes. In general, try to avoid adding axioms - it's better to have admitted theorems that will be (hopefully) eventually proved.
- Strongly prioritize properly finishing/debugging all proper definitions and (stating/proving) theorems from *all* sections of topology.tex . Doing the exercises is not as time-critical (although it may be occasionally needed as a prerequisite and generally useful, so use your judgement when to work on those admits)

PROGRESS MONITORING
- Keep numbered PROGRESSXXXX files. For each section (i.e. starting with 12 and ending with 50) monitor the progress (stubs only, definitions/theorems properly stated, theorems fully proved, section mostly/fully completed, exercises mostly/fully completed). I.e., this PROGRESS file has to have a statusline/paragraph for each of these 39 sections, which will be gradually changing towards more advanced levels.

- Make sure (and record in PROGRESSXXXX) that you make significant progress towards completing **ALL** (i.e. also later) sections. Dont get stuck only in the early sections. Use PROGRESSXXXX to find the imbalances and refocus on neglected parts (within means - dont jump around like crazy when working on something).

Dependency/Admit Status/Monitoring
- after each numbered backup, run `mgdeps6.pl bckXXXX > DEPSXXXX`
- this will create DEPSXXXX file with one (topology section) theorem per line with info as follows `topology_elem_subset: lines:12,admit:NO, recadmit:NO, deps(2):[topology_on:D,topology_sub_Power:T].`
- this says if the theorem is (recursively) admitted and its direct topology deps (D for a def and T for theorem). (note that the lines info there is after deleting all comments and that all non-topology deps are ignored)
- Use this to monitor the progress and find admitted theorems (major bottlenecks) that have to be proved so that others are recursively admit-free.

Compilation Checking
- **Run megalodon frequently** to check for compilation errors
- **When megalodon produces no output, the code is correct**
- **Megalodon only prints output when there are errors or warnings**
- Silent output = success – After any significant change, run: `timeout 30 megalodon -owned ownedSep132023 Math_Background.mg`- Often do numbered backups like bck1107
- With each numbered backup, also write the numbered summary changes file like CHANGES1107 (it should really be a summary, not just a simple diff).
- You can lookup your previous work in these CHANGES files when unsure how to continue.
- Never overwrite an older backup file. The numbering has to continue from the latest number. You must find it by running in tst7: ls bck* | sed 's/[^0-9]*//g'

```
| sort -n | tail -n 1.

## Why This Rule Exists
- Work before the topology section may have external dependencies
- Topology section is self-contained and safe to modify
- Breaking this rule wastes time undoing changes

## Last Updated
2025-12-02: Initial creation after accidentally editing lines 1728-1761
```

3.1 提示自动化

12月22日之前，我一直在“照看”命令行界面，也就是说，时不时查看智能体是否已完成工作并等待新的提示。这还算应付得来，因为我早期每月20美元的大语言模型订阅很快就会用完额度，而大部分时间都只能等待每周额度上限重置。

有了每月200美元的 ChatGPT Pro 订阅后，使用额度大幅提高，提示自动化也变得必不可少。我编写了以下脚本来实现这一点：监视 tmux 会话；如果会话连续60秒没有任何变化，就向其中输入提示，然后按下“Enter”：

```
#!/usr/bin/env bash

OUTDIR="/home/urban/ec/claude-code/tmux"
INTERVAL=60
mkdir -p "$OUTDIR"

while true; do
  ts=$(date +"%
  tmux capture-pane -p -S -1 > "$OUTDIR/capture_$ts.txt"
  # get two most recent captures
  mapfile -t files < <(ls -1t "$OUTDIR"/capture_*.txt 2>/dev/null | head -n 2)
  if [[ ${#files[@]} -eq 2 ]]; then
    if diff -q "${files[0]}" "${files[1]}" >/dev/null; then
      tmux send-keys 'Read the file CLAUDE.md . Treat it as authoritative work instructions.
Follow those instructions exactly for all subsequent actions and responses. That means work
as long as possible (as specified) without stopping.'; sleep 2; tmux send-keys Enter
    fi
  fi
  sleep "$INTERVAL"
done
```

4 其他提示词

4.1 初始

起初，我做了大量提示方面的实验，也针对一些特殊任务设计了专门的提示，例如初始创建与教科书对应的所有定义和定理/引理。以下是 11 月份要求 Codex 进行拓扑研究的前五条提示：

1. Now that you have learned the basics we will do something more serious. First read the file topology.tex . Then start formalizing it in Megalodon. The

```
background from which should learn the syntax of the definition, theorem and
proofs and build on it is in the tst7/Math_Background.mg . Here is how you
verify the article written so far: megalodon -owned ownedSep132023 Math_Background.mg
. Always make numbered backups of the .mg files as before. Do 5 attempts
and then stop.
2. Good, please continue. 5 more attempts.

3. Good. But maybe don't focus on the proofs so much and instead push forward
to cover all the definitions and toplevel theorems/lemmas from the tex file.
5 more attempts.
4. Good. But maybe don't focus on the proofs so much and instead push forward
to cover all the definitions and toplevel theorems/lemmas from the tex file.
5 more attempts. Try to do more without waiting for me, especially when things
compile. Feel free to do the whole tex file at once.
5. Good. Now let's stop for a moment with the formalization. Instead, put into
comments before each current definition and theorem the name, number, pagenumber,
location and/or statement from the original tex files, so that the
reader can easily check that the from/informal items correspond to each other.
Note that the comment syntax is as follows: (** from Part 1 **) .
```

一项重要的初始工作，是尽可能让编码智能体独立运行，并尽可能延长运行时间，避免它中途返回、提出额外问题，或以各种方式偏离方向。最终，我开始能让它独立运行一个多小时，有 Pro 订阅时甚至能运行两个小时，然后才给它下一个提示（而下一个提示通常只是重复同一个提示，直到我像上文所述那样将整个过程完全自动化）。

4.2 聚焦提示

12月31日，系统已经在乌雷松引理上做了不少工作，但随后开始转而处理其他事情。由于这看起来像是一个有趣的年终挑战，我发出了以下特别提示：

```
Special rule for today: Return to the proof of Urysohn lemma and finish it. Don't
switch to other things before it is fully finished without any admits, axioms,
etc. Finishing it is your singular goal for now.
```

这有点适得其反，因为系统随后开始尝试各种通常被禁止的技巧。我告诉它继续遵守 CLAUDE.md 中的规则，但要专注于完成乌雷松引理。最终，这个 3000 行的证明确实在2025年年底完成了。

1月4日，在系统对 Tietze 延拓定理做出一次重要（但很可能是多余的——见下文）初始尝试后，开始转向其他内容时，我也进行了一次类似的聚焦尝试。由于这件事很有意思，我尝试按如下方式聚焦（这次我更明智，还告诉它要遵守规则——效果不错）：

```
Read the file CLAUDE.md . Treat it as authoritative work instructions. Follow
those instructions exactly for all subsequent actions and responses. That means
work as long as possible (as specified) without stopping. Special rule for today:
Return to the proof of Tietze_stepII_real_extension_nonempty and finish it. Don't
switch to other things before it is fully finished.
```

这确实促成了一项大规模工作，在大约 20 小时后完成了一万行证明（以及其所有前置要求）。我认为，这现在提供了一种相当实用的方式，将系统对大型（教科书）形式化的自动化工作，与用户在希望优先完成某个特定证明时偶尔不耐烦地进行干预结合起来。

5 更多实用工具

5.1 Hammer（aby）尝试

自 2024 年以来，Megalodon 提供了一个简单的高阶自动证明器“aby” [1]。具体来说，它可以将问题导出为 TH0 TPTP 格式，并调用例如高阶 Vampire。目前还没有证明重构功能（如 Naproche 中的功能）。最初，省去大语言模型撰写详细 Megalodon 证明的工作，尽可能使用自动证明器，似乎颇具吸引力。我编写了相关脚本来实现自动化（在一定程度上），但目前效果并不理想；这可能还需要进一步完善缓存和（至少是简单的）前提选择，也可能需要采用一些广为人知的一阶形式（FOF）转换方法。一个亟待解决的主要问题是，在缺少证明重构功能的情况下，当内容发生变化时，如何智能地重新运行自动定理证明器。

5.2 定理与依赖跟踪

在开发过程中相对较晚的时候，我希望大语言模型能更全面地了解定理依赖关系和最大的“瓶颈”。例如，实数的完备性似乎是递归 admit 的主要原因之一，尽管根据 Brown 的超现实数开发，这应该是相当可行的（事实证明确实如此——见下文的 R_lub_exists）。因此，我添加了一个简单的依赖关系跟踪脚本，每次迭代时都会针对当前文件运行，并为每个定理生成一行信息，列出其当前状态（已完全证明、已 admit、递归 admit）、（部分）证明的长度以及当前的证明依赖关系。这似乎帮助大语言模型更好地理清了思路（另见 CLAUDE.md 中关于此事的建议）。

6 规范工作顺序

这是一个相对未经探索的领域。进行数学形式化的人已经形成了良好的工作惯例，但据我所知，还没有一本关于“良好形式化实践”的手册。

出现的一些问题包括：

1. 大语言模型起初在形式化定理和定义陈述时，有时会偷懒。它们有时会创建占位定义、临时拼凑的特殊情况，而不是一般情况，等等。如果大语言模型之后忘了这一点（它的上下文窗口有限），并把这些简化后的规格当真，就会很危险。我通过提醒它对此类问题务必谨慎来应对，有一两次甚至专门编写提示词，让它评估并消除这些不准确之处。2. 起初，它倾向于先处理简单的引理（通常是练习或示例），而不是困难或篇幅较长的定理。此外，它还可能没完没了地四处寻找简单的问题。关于我为防止这种情况所做的尝试（同时避免过于严格），请参见 CLAUDE.md。3. 我还希望它能持续关注所有章节，而不是长时间只埋头于某一节，因为这之后可能会导致大规模重构。我曾试图说服它维护一份 PROGRESS（进度）文件（见 CLAUDE.md），但这一

尝试迄今效果好坏参半。在深入研究并证明困难定理，与掌握整体形式化进展之间做出权衡，即使在大规模人工形式化项目中显然也并非易事。

7 形式化增长

表 1 展示了每 100 次备份（对应提交）后的行数。可以看到，切换到 ChatGPT Pro 后，增长速度加快了。

提交次数 → 行数							
提交	行	提交	行	提交	行	提交	行
100	1364	1000	7565	1900	44529	2800	78777
200	2033	1100	9919	2000	47542	2900	83890
300	3012	1200	11737	2100	52695	3000	90028
400	3523	1300	12620	2200	56411	3100	96056
500	3875	1400	13687	2300	56210	3200	105042
600	4447	1500	19891	2400	62531	3300	116433
700	4500	1600	31275	2500	68110	3400	119139
800	4561	1700	31401	2600	72784	3500	128287
900	4850	1800	39832	2700	68936	3600	137231
						3700	148283

表1 形式化的增长：提交次数与行数

8 所用资源

虽然 Pro 订阅确实只有每月 200 美元，但我尝试使用独立工具来衡量调用大语言模型的美元成本。具体来说，我每小时调用一次 `ccusage` 工具。表2 显示了运行三天的报告成本，其中包括一次与压缩相关的重启。请注意，这些数字远高于两周内花费的 100 美元。不禁让人思考，大语言模型公司究竟采取了什么样的商业政策（为争夺市场/订阅用户而展开恶性竞争？向企业收取过高的 API使用量费用？试图尽可能获取更多数据？诱饵式营销？等等）。

1–12	13–24	25–36	37–48	49–60	61–72
1918	2096	2229	2367	2549	602
1932	2112	2234	2378	2565	618
1949	2127	2239	2394	2576	635
1964	2145	2243	2409	2576	651
1980	2163	2254	2427	2576	664
1993	2177	2268	2442	2576	678
2008	2192	2282	2459	2576	694
2024	2206	2296	2475	2576	707
2040	2210	2312	2488	475	
2053	2215	2326	2503	489	
2067	2220	2339	2518	506	
2080	2224	2354	2534	519	

表2 连续的每小时成本报告（第1–72小时），按6列排列，每列12小时

遇到使用限额

在半自动化的早期模式下，我很快就用完了每月 20 美元的 Codex 或 Claude Code 订阅所包含的使用额度。不过，在这些非常廉价的订阅中，前 3 万行大约用了一个月生成。这对许多人来说可能已经非常有用了。我使用每月 200 美元的 Pro 订阅已经两周，并且每周都会遇到使用限额。第一周，额度几乎撑完整周，只在大约半天的时间里用尽。第二周，每周额度更早耗尽了（在1月5日，也就是额度重置前 3 天），而此前大约 8 小时刚完成下文所述的“Tietze Hill 之战”。可能有多种因素在影响这一点。

9 杂项注释

总的来说，我目前没有时间和精力对哪些方面做得好、哪些方面出了问题进行全面分析。今后可以利用大语言模型和其他工具，从完整日志（我一直在保存）以及公开的 Git 提交中自动找出有意思的案例。[11] 或许，这类分析可以形成反馈循环，用于改进目前使用的提示词和工具。

9.1 上下文压缩

这是我使用的基于大语言模型的技术面临的一个重大问题。时不时地，编码智能体需要从其有限的上下文窗口中删去不必要的部分，只保留后续处理必不可少的内容。压缩的结果（其专有性很可能不亚于我使用的大语言模型）可能相当难以预料。

更糟的是，连续会话能够进行的压缩量似乎存在硬性上限。我已经两次触及这个上限：一次是在12月26日，当时 ChatGPT 会话文件大小约为 252MB；另一次是在1月2日，会话文件大小达到 351MB。触及这个硬性上限后，会话似乎就无法以任何方式继续了。我找到的常见建议似乎是“不要运行长会话”。相反，在进行这类较长的项目时，人们应该自行维护关键状态和进度，并始终通过新会话载入更新后的状态和进度。

由于这看起来是个规模更大的项目，而且我喜欢长时间运行的会话的表现（不想丢失它学到的所有内容），所以我决定无视这条建议。我没有彻底重置，而是将会话回退到较早的版本。具体来说，触及硬性上限时，我只需用一个（小得多、也）早得多的版本，重写（很大的）ChatGPT 会话（jsonl）文件。两次重置时，我都使用了12月18日的早期版本会话文件，大小为 65MB。这种做法可能不受支持，也很危险，但到目前为止对我来说效果相当不错。

9.2 运行 Megalodon

随着文件不断增大，运行 Megalodon 会变得越来越慢，也可能出现问题。最初的库文件只需不到 1 秒即可运行完成，而我在笔记本电脑上进行的最新验证则需要 32 秒。不过，该文件现在已接近 17 万行，大小达 8 MB：

```
mptp@ar-2:~/ec/mizwrk/tst7$ time /home/mptp/ec/repos-mws/mgw_test/bin/megalodon bck4003
real 0m32.251s
```

[11] https://github.com/mgwiki/mgw_test/commits/main/

```
user 0m32.095s
sys 0m0.154s
mptp@ar-2:~/ec/mizwrk/tst7$ wc bck4003
 169122 1241687 7807911 bck4003
```

一个显而易见的解决方案是拆分文件，或者开始对已完成的定理使用 admit（Mizar 中的 `@proof` pragma）。到目前为止我还没有这么做，因为：（i）我希望大语言模型仍有可能重构这些证明；（ii）与大语言模型的使用和可用额度相比，运行 Megalodon 似乎并非瓶颈。也就是说，对大语言模型的调用次数仍然相当多，即使订阅了 Pro 订阅，我也会在使用限额重置前用完大语言模型的额度。

已证明的 10 个主要定理

表 3 表中列出了证明长度超过 300 行（不含 admit 或递归 admit）的已证明定理，并按其直接 Megalodon 证明的长度排序。我意识到，这可能遗漏了一些以较简单推论形式陈述的主要结果（例如见下一节），不过仍具有一定的参考价值。表中的定理名称可以点击，并链接至 mgwiki 中对应的 HTML 展示。

表 3 拓扑学定理（mgwiki 链接），包括行数和依赖数量。

定理	行	Deps
Tietze_stepII_real_extension_nonempty	10369	142
Urysohn_lemma	2964	85
Urysohn_metrization_theorem	2174	112
one_point_compactification_exists	1719	38
closed_interval_affine_equiv_minus1_1	1638	48
order_topology_basis_is_basis	1164	7
order_rel_trans	1159	22
unit_interval_connected	1049	32
uniform_topology_finer_than_product_and_coarser_than_box	940	74
convex_subspace_order_topology	931	26
ordered_square_not_subspace_dictionary	922	42
countability_axioms_subspace_product	911	51
product_topology_full_regular_axiom	907	56
rectangular_regions_basis_plane	894	12
interval_connected	878	23
completely_regular_product_topology	876	56
bounded_transform_psi_preimage_open_ray_upper	812	26
Romega_D_metric_open_ball_in_product_topology	805	62
ball_inside_rectangle	797	20
ex13_8b_halfopen_rational_basis_topology	768	18
Sorgenfrey_line_Lindelof	756	52
ex16_9_dictionary_equals_product	736	55
pointwise_limit_continuity_points_dense	729	58
ex13_8a_rational_intervals_basis_standard	710	16
ex16_10_compare_topologies_on_square	693	40
continuous_combine_or_01	677	43

XX：12 13万行形式拓扑，两周

定理	行	Deps
metrizable_spaces_normal	674	37
two_by_nat_singleton_not_in_basis	664	8
double_minus_one_map_continuous	629	43
embedding_via_functions	625	52
compact_space_net_has_accumulation_point	615	23
regular_countable_basis_normal	600	28
compact_space_net_has_accumulation_point_on	590	23
R_standard_basis_is_basis_local	585	9
euclidean_spaces_second_countable	581	48
preimage_mul_fun_R_open_ray_upper_in_product_topology	579	42
Tietze_extension_open_interval	576	48
Romega_extend_map_continuous_in_Romega_product	574	47
product_countable_basis_at_point_if_components_first_countable	574	34
abs_Cauchy_sequence_converges_R_standard_topology	563	30
order_rel_trichotomy_or_impred	560	13
preimage_mul_fun_R_open_ray_lower_in_product_topology	551	41
R_upper_limit_basis_is_basis_local	534	10
Q_sqrt2_cut_not_interval_or_ray	533	31
abs_Cauchy_sequence_converges_R_standard_topology_early	531	30
Sorgenfrey_plane_not_normal	525	33
singleton_rectangle_in_dictionary	514	45
subnet_converges_to_accumulation	495	32
subnet_converges_to_accumulation_witnessed	487	32
open_ball_refine_intersection	481	17
continuous_construction_rules	475	16
ex17_5_basis_elem_meets_interval	474	10
flip_unit_interval_continuous	473	36
uniform_cauchy_metric_complete_imp_uniform_limit_stub	467	30
subnet_converges_to_accumulation_witnessed_on	465	32
A_N_eps_closed_stub	463	32
paracompact_Hausdorff_normal	460	28
Tietze_extension_minus1_1	457	48
product_sequence_convergence_iff_coordinates	456	42
R_lower_limit_basis_is_basis_local	454	9
unbounded_sequences_in_Romega_box_topology	437	27
preimage_add_fun_R_open_ray_upper_in_product_topology	435	32
ex30_4_compact_metrizable_second_countable	425	48
regular_normal_via_closure	421	14
limit_point_compact_not_necessarily_compact	421	25
separation_subspace_limit_points	419	9
paracompact_Hausdorff_regular	416	25
one_minus_fun_continuous	413	32
ex13_6_Rl_RK_not_comparable	411	24
double_map_continuous	411	41
distance_R2_triangle_sqr_Le	409	14
Cauchy_with_convergent_subsequence_converges	404	41
finite_product_compact	402	21

定理	行	Deps
Romega_product_cylinder_in_D_metric_topology	399	37
R_standard_plus_K_basis_is_basis_local	397	9
continuity_via_nets	383	33
Sorgenfrey_plane_L_closed	381	34
Heine_Borel_closed_bounded	380	26
closed_subspace_paracompact	378	16
R_lub_exists	376	13
continuity_via_sequences_metric	374	21
ex17_17_closure_A_C_Supq	370	19
Romega_singleton_map_continuous_prod	367	31
ex30_14_product_Lindelof_compact	362	28
product_finite_subbasis_intersection_separator	357	43
uniform_limit_of_continuous_to_metric_is_continuous	352	35
ex30_5a_metrizable_countable_dense_second_countable	347	39
ex23_connected_open_sets_path_connected	347	27
completely_regular_subspace_product	346	49
rectangle_inside_ball	345	25
preimage_add_fun_R_open_ray_lower_in_product_topology	341	32
path_component_transitive_axiom	335	41
compact_subspace_via_ambient_covers	335	8
locally_m_euclidean_implies_T1	330	25
ex30_5b_metrizable_Lindelof_second_countable	328	34
Romega_D_metric_induces_product_topology	327	46
product_basis_generates	322	10
U_eps_open_dense_stub	320	38
RK_not_regular_axiom	320	23
Sigma_countable	318	2
order_topology_interval_refine	313	5
topology_chain_four_sets	310	4
pasting_lemma	306	18
ex13_7_R_topology_containments	304	23
Sorgenfrey_plane_diag_rational_closed	303	33
finite_product_connected	303	17
compact_Hausdorff_normal	301	17

表4 展示了 ChatGPT 选出的 30 个主要定理/引理的评估。这些内容可能存在各种错误。这些定理/引理（以及其他一些定理/引理）的确切表述，以 LaTeX 和 Megalodon 格式列于附录中。

表4 ChatGPT 对 30 项主要结果的筛选与评估

节与编号 (topology.tex)	**定理 / 引理名称**	**形式化条目**	**ChatGPT 对形式化的评估**
第15节，定理15.1	积拓扑的基 拓扑学	product_basis_ generates_product_ topology	基论证的清晰抽象 忠实于教科书证明， 很好地复用了基引理。

XX：14 13万行形式拓扑，两周

节与编号	定理 / 引理	形式化条目 (:T)	评估
第16节，定理16.3	子空间 of a 积	product_subspace_topology	对诱导拓扑的处理明确而稳健。
§18，定理 18.3	粘贴引理	pasting_lemma	对连续性进行出色的压力测试——基础设施；假设范围明确——界定清楚。
§18，定理 18.4	映入积空间的映射	maps_into_products	非常标准的命题，形式化——化过程与教科书高度一致。
§23，引理 23.1	通过分离性证明连通性	connected_space_no_separation	等价性表述正确；略微偏底层的集合推理，但可靠。
§23，引理 23.2	连通子集与分离	connected_subset_in_separation_side	忠实编码；可复用引理供后续证明使用。
§23，定理 23.3	连通集的并 sets	union_connected_common_point	良好的抽象，证明结构符合标准论证。
§23，定理23.5	连续空间的连续像连通空间	continuous_image_connected	典范定理；形式化陈述简洁且通用。
§23，定理23.6	有限乘积的连通空间	finite_product_connected	正确 归纳法 on 有限乘积； 抽象 层级恰当。
第26节，定理 26.2	紧空间的闭子空间紧空间	closed_subspace_compact	非常标准；为.....奠定了良好基础后续的紧致性结果。
第26节，定理 26.3	紧子集 of 豪斯多夫空间	compact_subspace_in_Hausdorff_closed	忠实的分离论证；预-表明选择得当。
第26节，定理26.5	紧空间的连续像紧空间	continuous_image_compact	核心引理；结构良好且可重复使用。
第26节，定理26.6	紧致–豪斯多夫双射	compact_to_Hausdorff_bijection_homeomorphism	表述极佳；拓扑上典范的。
第26节，定理26.7	有限乘积紧空间	finite_product_compact	非平凡的归纳法；证明基础设施——结构看起来很扎实。
第26节，引理26.8	管引理	tube_lemma	对积邻域的处理得当，忠实于 Munkres。
第27节，定理27.3	海涅–博雷尔 Theorem	Heine_Borel_closed_bounded	重要结果；假设条件和等价关系编码正确。
第27节，引理 27.5	勒贝格 数值引理	Lebesgue_number_lemma	良覆盖工具；略显冗长，但稳健。
第27节，定理 27.6	一致连续性在紧空间上	uniform_continuity_on_compact	正确 度量-拓扑的相互作用；良好的抽象。
§30，定理 30.3	可数基 ⇒ 林德洛夫	countable_basis_implies_Lindelof	结构优美的可数性论证。
§30，定理 30.4	可数基 ⇒ 可分	countable_basis_implies_separable	忠实、简洁；良好复用基的工具。
§32，定理 32.1	正则 + 可数-基 ⇒ 正规	regular_countable_basis_normal	重要的分离性结果；依-赖关系恰当。
第32节，定理32.3	紧致豪斯多夫空间是正规的	compact_Hausdorff_normal	标准论证；形式化很简洁。

节 & 编号	定理 / 引理	形式化条目 (:T)	评估
第33节，定理 33.1	乌雷松引理	Urysohn_lemma	优秀：假设正确，非平凡构造处理得当。
第34节，定理 34.1	乌雷松 度量化定理	Urysohn_metrization_theorem	高价值成果；形式化看起来忠实且结构良好。高价值成果；形式化看起来忠实且结构良好。
第35节，定理35.1	Tietze 延拓定理	Tietze_extension_interval	非常强的结果；形式化陈述和依赖关系看起来合理。ment 和依赖关系看起来合理。
第41节，定理41.1	仿紧 Hausdorff ⇒正规	paracompact_Hausdorff_normal	局部有限覆盖的良好抽象。
第41节，定理41.2	……的闭子空间仿紧空间	closed_subspace_paracompact	正确且可复用；集成良好——完成。
第41节，引理41.3	Michael 引理	Michael_lemma_41_3	技术上 要求很高； 扎实形式化处理。
第41节，引理 41.6	收缩引理	shrinking_lemma_41 6_	对细化的控制良好；抽象也足够恰当。
§48，引理 48.1	贝尔空间引理	Baire_space_dense_Gdelta	正确使用可数交；为范畴理论打下坚实基础参数。

11 Tietze 山之战

过去两天的大部分时间都花在形式化 Tietze 延拓定理上，具体来说是第二步。其（自动形式化的）Megalodon 陈述如下：

```
(** helper for §35 Step II: nonempty closed subset case, real-valued extension **)
(** LATEX VERSION: Step II (nonempty A): construct a real-valued continuous
extension gR:X->R agreeing with f on A and bounded in [-1,1]. **)
Theorem Tietze_stepII_real_extension_nonempty : forall X Tx A f:set,normal_space X Tx ->
closed in X Tx A ->A <> Empty ->continuous_map A (subspace_topology X Tx A)
(closed_interval (minus_SNo 1) 1) (closed_interval_topology (minus_SNo 1) 1) f ->
exists gR:set,continuous_map X Tx R R_standard_topology gR /\
(forall x:set, x :e A -> apply_fun gR x = apply_fun f x) /\
(forall x:set, x :e X -> apply_fun gR x :e closed_interval (minus_SNo 1) 1).
```

证明于1月4日20:15完成，并附上了以下（考虑到付出的努力，内容相当枯燥的）关于备份编号3893的进度消息：

```
Update 3893: §35 Step II: eliminated the temporary admit in Htail_budget by adding
the missing `HgN0xS` (for the current x0) and closing the final SNoLe_tra transitivity;
Tietze_stepII_real_extension_nonempty now has no admits.
```

证明[12] 包含 10369 行，有 142 个直接依赖项（36 个定义和 106 个定理）。

[12] https://mgwiki.github.io/mgw_test/topology.mg.html#Tietze_stepII_real_extension_nonempty ，也是 https://raw.githubusercontent.com/mgwiki/mgw_test/48bca1fc5df4576f1a28416aa4e423ecfa556f6c/mglib/topology.mg的第129871行

XX:16 13万行形式拓扑 两周

教科书证明用到了关于实值函数的无穷级数及其一致收敛的事实，这是一项教科书中没有涉及的主要外部依赖。智能体似乎被这个问题难住了，于是转而将归纳构造（无穷构造）不断展开了 12 次（即级数的 12 项），生成了约 6000 行内容；这些内容对最终证明没有帮助（尽管在完成归纳之前，它们可能有助于更具体地理解其中的机制）。

不过，它已经从中恢复过来，并决定逐步形式化必要的预备知识。关于那次未完成的 9000 行证明尝试，ChatGPT 的分析可在线查看[13]。以下是其定期报告，展示了大语言模型如何因教科书中要到很后面才讲完备度量空间这一事实而感到困惑。

```
CHANGES3682:
Attempted refactor to move the uniform-limit/uniform-Cauchy
infrastructure (continuous_at_map, pointwise_limit_metric,
uniform_limit_metric, uniform_cauchy_metric, and the uniform-limit
continuity lemmas) earlier near §35 to support Tietze Step II.

Result:
- This does not work because that block depends on
`complete_metric_space`, which is defined later (in the metric
completeness section), so moving it before §35 causes a
forward-reference error: "Unknown term complete_metric_space".

Action taken:
- Reverted Math_Background.mg back to bck3681 state (no content loss;
only the failed move was undone).
```

具体来说，问题在于，完备度量空间及其中的柯西序列确实只在 Munkres 的第 43 节才定义，而 Tietze 定理则在第 35 节。大语言模型一直努力遵循形式化规则。因此，它进行了 11 次毫无用处的迭代和修正（从 CHANGES3650 中的 u2 到 CHANGES3685 中的 u12），在这 35 次改动中，证明从 865 行膨胀到了 5408 行。这耗费了将近 4 小时，也浪费了不少资源。经过几次徒劳的尝试后，大语言模型终于说服自己先处理预备知识这项艰巨工作，并将其移到 Tietze 定理之前，具体如下：

```
CHANGES3687:
Move uniform-limit infrastructure before §35 (compiles)
============================================================

Goal:
- §35 (Tietze extension theorem, Step II) needs uniform-limit /
completeness tools without forward-reference blockers.

What changed:
- Moved the block starting at "helper: continuity at a point into a
topological space" (continuous_at_map, pointwise_limit_metric,
uniform_limit_metric, uniform_cauchy_metric, uniform-limit continuitylemmas,
 plus related helpers) from its later location to the area justbefore §35.
- Kept §48-specific helper material (A_N_eps, U_eps,U_eps_open_dense_stub,
 etc.) located near §48 again,  so it stays afterthe Baire_space definitions.
```

[13] https://chatgpt.com/share/695a7928-9efc-8002-860f-bb2c0f33663b

```
- Added early stubs needed by the moved uniform-limit block:
  - metric_topology_R_bounded_metric_eq_R_standard_topology_early
  - R_bounded_metric_is_metric_on_early
  These avoid forward references to the later bounded-metric
  development; they should later be replaced by the proved theorems.
- Left a note in the original location indicating the helper block was moved earlier.
```

请注意，这属于较大的间隔、外部依赖，甚至是教科书中的（不安全）前向引用——这可能会给人工形式化带来严重障碍。关于最终的一万行证明及其与此前未完成的九千行证明之间差异的 ChatGPT 分析已在网上发布[14]。这场艰苦的战斗结束后，Munkres 对 Tietze 定理的标准表述如下：

定理 35.1（Tietze 延拓定理）。设 X 是正规空间；设 A 是 X 的闭子空间。

(a) 从 A 到 $\mathbb{R}$ 的闭区间 $[a,b]$ 的任意连续映射，都可以延拓为从整个 X 到 $[a,b]$ 的连续映射。

(b) 从 A 到 $\mathbb{R}$ 的任意连续映射，都可以延拓为从整个 X 到 $\mathbb{R}$ 的连续映射 .

可通过 Megalodon 的这两个定理（相对容易地）证明：

```
Theorem Tietze_extension_real_bounded_interval : forall X Tx A a b f:set,
Rle a b ->normal_space X Tx -> closed_in X Tx A ->
continuous_map A (subspace_topology X Tx A) R R_standard_topology f ->
(forall x:set, x :e A -> apply_fun f x :e closed_interval a b) ->
exists g:set, continuous_map X Tx R R_standard_topology g /\
(forall x:set, x :e A -> apply_fun g x = apply_fun f x).

Theorem Tietze_extension_real : forall X Tx A f:set,
normal_space X Tx -> closed_in X Tx A ->
continuous_map A (subspace_topology X Tx A) R R_standard_topology f ->
exists g:set, continuous_map X Tx R R_standard_topology g /\
(forall x:set, x :e A -> apply_fun g x = apply_fun f x).
```

12 结论

这显然相当令人惊讶，也令人振奋，我很好奇这一切最终会走向何方。这里的成本（以及我投入的编码工作）都非常低。2026 年，我们很有可能会看到大部分（撰写得还算规范的）数学教科书和论文都实现自动形式化。也可能不会——这件事最终或许会遇到局限。或许，对某些（请在此填入一个流行的证明助手名称）原教旨主义者而言，一个（迫切需要的？）现实检验是：目前这些实验使用的是一款几乎无人知晓、基于高阶集合论的证明助手，而且几乎没有接触现有的大语言模型。AI 辅助（自动）形式化的未来似乎相当开放，也相当民主。

[14] https://chatgpt.com/share/695a7928-9efc-8002-860f-bb2c0f33663b

13 致谢

Chad Brown 主要开发了 Megalodon、它的 hammer 以及初始的核心集合论库；他还帮我为实验搭建环境，并对早期系统的形式化成果给予了鼓励性的评价。Zar Goertzel 一直向我介绍他使用大语言模型以及 Claude Code、Codex 等相关编码智能体所做的实验。他报告了令人鼓舞的结果后，我听从了他的建议，全面改用 ChatGPT 5.2（以及每月 200 美元的 Pro 订阅）。他已经完成了 Ramsey(3,6) 定理的自动形式化，共 1.3 万行。Atle Hahn 和 Adrian De Lon 一直在 Naproche 中进行有意思的大语言模型自动形式化实验。Atle 令人印象深刻且确实有效的提示词，展现了通过“编程”让大语言模型进行自动形式化的力量与潜力。感谢 John Harrison 对我们早期自动形式化工作的支持，也感谢 Larry Paulson 和 Dominic Mulligan 的鼓励。Cezary Kaliszyk 协助开发 Megalodon/hammer 及相关基础设施，并从 2014 年起与我一道率先探索基于学习的自动形式化理念、项目和方法。自那以后，自动形式化社区发展壮大了许多——感谢所有这些（尤其是早期的）支持者。

参考文献

1 Chad E. Brown、Cezary Kaliszyk、Martin Suda 和 Josef Urban。高阶集合论的 hammer 方法。载于 Valeria de Paiva 和 Peter Koepke 编，智能计算机数学——第18届国际会议，CICM 2025，巴西利亚，巴西，2025年10月6日至10日，论文集，《计算机科学讲义》第16136卷，第3–20页。施普林格，2025年。 doi:10.1007/978-3-032-07021-0_1。2 David Cariolaro。关于 Ramsey 数 R (3, 6)。奥尔堡大学数学科学系，1999年。3 Thibault Gauthier、Miroslav Olsák 和 Josef Urban。异体编码。《国际近似推理杂志》，162:109009，2023年。URL: https://doi.org/10.1016/j.ijar.2023.109009, doi:10.1016/J.IJAR.2023.109009。4 Thibault Gauthier 和 Josef Urban。从头学习猜想生成。载于 Clark W. Barrett 和 Uwe Waldmann 编，自动演绎——CADE 30——第30届自动演绎国际会议，德国斯图加特，2025年7月28日至31日，论文集，《计算机科学讲义》第15943卷，第423–445页。施普林格，2025年。 doi:10.1007/978-3-031-99984-0_23。5 Jan Jakubuv、Karel Chvalovský、Zarathustra Amadeus Goertzel、Cezary Kaliszyk、Mirek Olsák、Bartosz Piotrowski、Stephan Schulz、Martin Suda 和 Josef Urban。Mizar 60，为 Mizar 50 献礼。载于 Adam Naumowicz 和 René Thiemann 编，第14届交互式定理证明国际会议，ITP 2023，波兰比亚韦斯托克，2023年7月31日至8月4日，《LIPIcs》第268卷，第19:1–19:22页。Schloss Dagstuhl - Leibniz-Zentrum für Informatik，2023年。URL: https://doi.org/10.4230/LIPIcs.ITP.2023.19, doi:10.4230/LIPICS.ITP.2023.19。6 Cezary Kaliszyk、Josef Urban 和 Jirí Vyskocil。在对齐语料库上学习解析（粗略钻石）。载于 Christian Urban 和 Xingyuan Zhang 编，交互式定理证明——第6届国际会议，ITP 2015，中国南京，2015年8月24日至27日，论文集，《计算机科学讲义》第9236卷，第227–233页。施普林格，2015年。 doi:10.1007/978-3-319-22102-1_15。7 Cezary Kaliszyk、Josef Urban、Jirí Vyskocil 和 Herman Geuvers。开发非形式化数学与形式化数学之间基于语料库的翻译方法：项目描述。载于 Stephen M. Watt、James H. Davenport、Alan P. Sexton、Petr Sojka 和 Josef Urban 编，智能计算机数学——国际会议，CICM 2014，葡萄牙科英布拉，

*2014*年7月7日至*11*日。论文集，《计算机科学讲义》，第435–439页。施普林格，2014年。doi:10.1007/978-3-319-08434-3_34。8 Jelle Piepenbrock、Josef Urban、Konstantin Korovin、Miroslav Olsák、Tom Heskes 和 Mikolás Janota。用于学习实例化证明中项综合的不变神经架构。《符号计算杂志》，128:102375，2025年。URL: https://doi.org/10.1016/j.jsc.2024.102375，doi:10.1016/J.JSC.2024.102375。9 Josef Urban。MaLARea：大型理论自动推理的元系统。载于 Geoff Sutcliffe、Josef Urban 和 Stephan Schulz 编，大型理论实证成功自动推理研讨会论文集，德国不来梅，*2007*年7月*17*日，第257卷，*CEUR*研讨会论文集。CEUR-WS.org，2007年。URL： https://ceur-ws.org/Vol-257/05_Urban.pdf。10 Josef Urban。面向 Mizar 的自动推理：通过知识交换实现人工智能。载于 Piotr Rudnicki、Geoff Sutcliffe、Boris Konev、Renate A. Schmidt 和 Stephan Schulz 编，*LPAR 2008*研讨会论文集：知识交换：自动定理证明器与证明助手，以及第七届逻辑实现国际研讨会，卡塔尔多哈，*2008*年*11*月*22*日，第418卷，*CEUR*研讨会论文集。CEUR-WS.org，2008年。URL： https://ceur-ws.org/Vol-418/paper1.pdf。

11 Josef Urban 和 Jan Jakubuv . 首批神经猜想数据集与实验 . 在 Christoph Benzmüller 和 Bruce R. Miller 编著的《智能计算机数学——第*13*届国际会议，*CICM2020*，意大利贝尔蒂诺罗，*2020*年7月*26*日至*31*日，会议论文集》，《计算机科学讲义》第12236卷，《计算机科学讲义》，第315–323页。施普林格，2020年。 doi:10.1007/978-3-030-53518-6_24。

12 Qingxiang Wang、Cezary Kaliszyk 和 Josef Urban . 首次尝试将非形式化数学神经翻译为形式化数学。载于 Florian Rabe、William M. Farmer、Grant O. Passmore 和 Abdou Youssef 编，《智能计算机数学——第*11*届国际会议，*CICM 2018*，奥地利哈根贝格，*2018*年*8*月*13*日至*17*日，论文集》，《计算机科学讲义》第11006卷，《计算机科学讲义》，第255–270页。施普林格，2018年。 doi:10.1007/978-3-319-96812-4\22。_

LaTeX 和 Megalodon 中主要定理一览

No.	topology.tex 中的陈述	正式名称	正式陈述（统计数据，精确）
定理 15.1	定理 15.1. 若 B 是 X 的拓扑的一个基，且 $\mathcal{C}$ 是一个基对于 Y 的拓扑，则集合族 $\mathcal{B} \times \mathcal{C} = \{B \times C \mid B \in \mathcal{B}$ 和 $C \in \mathcal{C}\}$ 是一个基对于 $X \times Y$ 的拓扑。	product_basis_generates_product_topology	forall X Y Bx By:set, basis_on X Bx -> basis_on Y By -> product_topology X (generated_topology X Bx) Y (generated_topology Y By) = generated_topology (setprod X Y) (product_basis_from X Y Bx By)
定理 16.3	定理 16.3. 若 A 是一个子空间属于 X，且 B 是 Y 的子空间，则积拓扑在 $A \times B$ 上与拓扑作为 $X \times Y$ 的子空间继承。	product_subspace_topology	forall X Tx Y Ty A B:set, topology_on X Tx -> topology_on Y Ty -> product_topology A (subspace_topology X Tx A) B (subspace_topology Y Ty B) = subspace_topology (setprod X Y) (product_topology X Tx Y Ty) (setprod A B)

XX：20 13万行形式拓扑两周

No.	topology.tex 中的陈述	正式名称	形式陈述（统计数据，精确）
定理 18.3	定理 18.3（粘贴 引理）。假设 $X = A \cup B$，其中 A 和 B 在 X。设 $f : X \to Y$ 是一个映射使得 $f \mid A$ 和 $f \mid B$ 是连续。如果 $f \mid A \cap B$ 是连续的——续，则 f 是连续的。	pasting_lemma	forall X Tx Y Ty A B f:set, top」ology_on X Tx -> topology_on Y Ty -> closed_in X Tx A -> closed_in X Tx B -> A :\/: B = X -> continuou」s_map A (subspace_topology X Tx A) Y Ty (restriction_map A f) -> co」ntinuous_map B (subspace_topology X Tx B) Y Ty (restriction_map B f) -> (forall x:set, x :e (A :/\: B) -> apply_fun f x = apply_fun f x) -> continuous_map X Tx Y Ty f
定理 18.4	定理 18.4。设 $f : X \to Y \times Z$ 由以下方程给出 $f(x) = (f_1(x)\ f_2(x))$。然后 f 当且仅当 f_1 连续并且 f_2 连续。	maps_into_products	forall A Ta X Tx Y Ty h:set, t」opology_on A Ta -> topology_on X Tx -> topology_on Y Ty -> (co」ntinuous_map A Ta (setprod X Y) (product_topology X Tx Y Ty) h <-> (continuous_map A Ta X Tx (projec」tion_map1 X Y o h) /\ continuous_」map A Ta Y Ty (projection_map2 X Y o h)))
引理 23.2	引理 23.2。如果 C 是一个连通 X 的子空间，并且与其相交分离 A 的两个集合 A 和 B 离 $X = A \cup B$，则 C 与 $A \cap B$ 相交。	connected_subset_in_separation_side	forall X Tx C D A:set, topolo」gy_on X Tx -> connected_space C (subspace_topology X Tx C) -> s」eparation_of X Tx A D -> C :/\: A <> Empty -> C :/\: D <> Empty -> C :/\: A :/\: D <> Empty
定理 23.3	定理 23.3。一个列具有公共点的连通子空间的集合具有公共点的 X 是连通的。	union_connected_common_point	forall X Tx F x0:set, topology_on X Tx -> (forall C:set, C :e F -> connected_space C (subspace_topo」logy X Tx C)) -> (forall C:set, C :e F -> x0 :e C) -> connected_sp」ace (Union F) (subspace_topology X Tx (Union F))
定理 23.5	定理 23.5。连通子空间的像连通空间在连续映射下仍是连通的。	continuous_image_connected	forall X Tx Y Ty f:set, topol」ogy_on X Tx -> topology_on Y Ty -> continuous_map X Tx Y Ty f -> connected_space X Tx -> connec」ted_space (image_of X f) (sub」space_topology Y Ty (image_of X f))
定理 23.6	定理 23.6。 有限个空间的积有限个连通空间是连通的。	finite_product_connected	forall X Tx Y Ty:set, connected_」space X Tx -> connected_space Y Ty -> connected_space (setprod X Y) (product_topology X Tx Y Ty)
定理 26.2	定理 26.2。 每个闭集子-紧空间的空间是紧致。	closed_subspace_compact	forall X Tx Y:set, compact_space X Tx -> closed_in X Tx Y -> compac」t_space Y (subspace_topology X Tx Y)
定理 26.3	定理 26.3。每个紧致子-豪斯多夫空间的空间是闭集。	compact_subspace_in_Hausdorff_closed	forall X Tx Y:set, Hausdorff_space X Tx -> compact_space Y (subspac」e_topology X Tx Y) -> closed_in X Tx Y

No.	topology.tex中的陈述	正式名称	正式陈述（统计信息，精确）
定理 26.5	定理 26.5。紧致空间在连续映射下的像是紧致的。	continuous_image_compact	forall X Tx Y Ty f:set, compact_space X Tx -> topology_on Y Ty -> continuous_map X Tx Y Ty f -> compact_space (image_of X f) (subspace_topology Y Ty (image_of X f))
定理 26.6	定理 26.6。设 $f: X \to Y$ 是一个双射连续函数。如果 X 是紧致的，且 Y 是豪斯多夫空间，那么 f 是一个同胚ism.	compact_to_Hausdorff_bijection_homeomorphism	forall X Tx Y Ty f:set, compact_space X Tx -> Hausdorff_space Y Ty -> bijection X Y f -> continuous_map X Tx Y Ty f -> homeomorphism X Tx Y Ty f
定理 26.7	定理 26.7。 有限个紧空间的积有限多个紧空间是是紧致的。	finite_product_compact	forall X Tx Y Ty:set, compact_space X Tx -> compact_space Y Ty -> compact_space (setprod X Y) (product_topology X Tx Y Ty)
引理 26.8	引理 26.8（管状引理）。考虑乘积空间 $X \times Y$，其中 Y 是紧致的。如果 N 是 $X \times Y$ 的一个开集，包含包含切片 $\{x_0\} \times Y$，那么存在 W 的一个邻域 x_0 中的某个集合，使得 $W \times Y \subset N$。	tube_lemma	forall X Tx Y Ty:set, topology_on X Tx -> topology_on Y Ty -> compact_space Y Ty -> forall x0 N:set, x0 :e X -> N :e product_topology X Tx Y Ty -> (setprod {x0} Y) c= N -> exists W:set, W :e Tx /\ x0 :e W /\ (setprod W Y) c= N
定理27.3	定理27.3（海涅–博雷尔定理）。 一个 A 的 $\mathbb{R}$子空间是紧致的，当且仅当它是在 $\mathbb{R}$ 中闭且有界。	Heine_Borel_closed_bounded	forall A:set, A c= R -> compact_space A (subspace_topology R R_standard_topology A) <-> (closed_in R R_standard_topology A /\ bounded_set A)
定理 28.1	定理 28.1。紧致性蕴含极限点紧致性，但反之则不然。	compact_implies_limit_point_compact	forall X Tx:set, compact_space X Tx -> limit_point_compact X Tx
定理 30.3	定理 30.3。假设 X 具有可数基。那么每个 X 的开覆盖都包含一个可数子集族，它覆盖 X。	countable_basis_implies_Lindelof	forall X Tx:set, second_countable_space X Tx -> Lindelof_space X Tx
定理 30.4	定理 30.4。如果 X 有一个可数的基，那么 X 包含一个可数稠密子集。	countable_basis_implies_separable	forall X Tx:set, second_countable_space X Tx -> separable_space X Tx
定理 32.1	定理 32.1。若 X 是正则的，并且有可数基，则 X 是正规的。	regular_countable_basis_normal	forall X Tx:set, regular_space X Tx -> second_countable_space X Tx -> normal_space X Tx
定理 32.3	定理 32.3. 每个紧致豪斯多夫空间都是正规空间。	compact_Hausdorff_normal	forall X Tx:set, compact_space X Tx -> Hausdorff_space X Tx -> normal_space X Tx

XX:22 13万行形式拓扑，两周

No.	topology.tex中的陈述	正式名称	正式陈述（统计数据，精确）
定理 33.1	定理 33.1（乌雷松引理）。设 X 是一个正规空间；设 A 且 B 是互不相交的闭集，属于 X。那么存在一个连续函数 $f: X \to [0,1]$，使得 $f(A) = \{0\}$ 以及 $f(B) = \{1\}$。	Urysohn_lemma	forall X Tx A B:set, normal_space X Tx -> closed_in X Tx A -> closed_in X Tx B -> A :/\: B = Empty -> exists f:set, continuous_map X Tx unit_interval unit_interval_topology f /\ (forall a:set, a :e A -> apply_fun f a = 0) /\ (forall b:set, b :e B -> apply_fun f b = 1)
定理 34.1	定理 34.1（乌雷松度量化定理）。每个正则空间具有可数基的可度量化。	Urysohn_metrization_theorem	forall X Tx:set, regular_space X Tx -> second_countable_space X Tx -> metrizable X Tx
定理 35.1	定理 35.1（Tietze 延拓定理）。 设 X 是一个正规空间，且 A 是一个闭子空间属于 X。那么每个连续映射 $f: A \to [-1,1]$ 都可扩展为一个连续映射 $F: X \to [-1,1]$。	Tietze_extension_interval	forall X Tx A f:set, normal_space X Tx -> closed_in X Tx A -> continuous_map A (subspace_topology X Tx A) closed_interval_minus1_1 closed_interval_minus1_1_topology f -> exists F:set, continuous_map X Tx closed_interval_minus1_1 closed_interval_minus1_1_topology F /\ forall a:set, a :e A -> apply_fun F a = apply_fun f a
定理 41.1	定理 41.1. 每个仿紧豪斯多夫空间都是正规空间。	paracompact_Hausdorff_normal	forall X Tx:set, paracompact_space X Tx -> Hausdorff_space X Tx -> normal_space X Tx
定理 41.2	定理 41.2。 任意闭子仿紧空间的空间都是仿紧的。	closed_subspace_paracompact	forall X Tx Y:set, paracompact_space X Tx -> closed_in X Tx Y -> paracompact_space Y (subspace_topology X Tx Y)
引理 41.3	引理 41.3（Michael 引理）。设 X 为正则。则每个 X 的开覆盖都有一个开局部有限细化，当且仅当它有一个开 σ-局部有限细化。	Michael_lemma_41 3_	forall X Tx:set, regular_space X Tx -> ((forall Ufam:set, open_cover X Tx Ufam -> exists Vfam:set, open_refinement X Tx Ufam Vfam /\ locally_finite_family X Tx Vfam) <-> (forall Ufam:set, open_cover X Tx Ufam -> exists Vfam:set, ᴼpen_refinement X Tx Ufam Vfam /\ sigma_locally_finite_family X Tx Vfam))
Th 41.5	定理 41.5。 每个 正则林德洛夫空间都是仿紧的。	regular_Lindelof_paracompact	forall X Tx:set, regular_space X Tx -> Lindelof_space X Tx -> paracompact_space X Tx
定理 46.10	定理 46.10。 设 X 是局部紧豪斯多夫空间。那么求值映射 $e: X \times C_k(X, Y) \to Y$ 是连续的。	evaluation_map_continuous	forall X Tx Y Ty:set, locally_compact_space X Tx -> Hausdorff_space X Tx -> topology_on Y Ty -> continuous_map (setprod X (Ck X Tx Y Ty)) (product_topology X Tx (Ck X Tx Y Ty) (compact_open_topology X Tx Y Ty)) Y Ty (evaluation_map X Y)

No.	topology.tex中的陈述	正式名称	正式陈述（统计数据，精确）
定理 22.2	定理 22.2. 设 $p : X \to Y$ 是一个商映射。如果 $A \subset X$ 是饱和的，那么限制 $p\|_A : A \to p(A)$是一个商映射 map.	restriction_ quotient_ saturated	forall X Tx Y Ty p A:set, quotien」 t_map X Tx Y Ty p -> saturated_set X p A -> quotient_map A (subspac」 e_topology X Tx A) (image_of A p) (subspace_topology Y Ty (image_of A p)) (restriction_map A p)
定理 17.11	定理 17.11. 每个简单序有序集合在实数集豪斯多夫中 恒一序拓扑。	simply_ordered_ set_Hausdorff	forall X Lt:set, simply_ordere」 d_set X Lt -> Hausdorff_space X (order_topology X Lt)
引理 23.1	引理 23.1. 若 X admit 一个分离，则 X 不连通 ted.	connected_ space_no_ separation	forall X Tx:set, topology_on X Tx -> (exists A B:set, separation_of X Tx A B) -> ~ connected_space X Tx
定理 23.4	定理 23.4. 设 A 是一个连通 X的子空间。若 $A \subset B \subset \overline{A}$, 则 B 是连通的。	connected_with_ limit_points	forall X Tx A B:set, topology_on X Tx -> connected_space A (subspac」 e_topology X Tx A) -> A c= B -> B c= closure_of X Tx A -> connecte」 d_space B (subspace_topology X Tx B)
定理 24.3	定理 24.3（介值定理）。设 $f : X \to Y$ 为连续，其中 X 是连-通且有序的，并且 Y 具有序拓扑。那么 $f(X)$ 是一个区间。	intermediate_ value_theorem	forall X Tx Y f a b r:set, simply」 _ordered_set X -> topology_on X Tx -> topology_on Y (order_topology Y) -> continuous_map X Tx Y (order_t」 opology Y) f -> between (apply_fun f a) r (apply_fun f b) -> exists c:set, c :e X /\ apply_fun f c = r
定理 25.1	定理 25.1。.....的连通分支空间 X 构成.....的一个划分 X 到连通子空间。	components_ partition_space	forall X Tx:set, topology_on X Tx -> partition_of X (components_of X Tx)
定理 25.5	定理 25.5。如果 X 是局部连-通的，那么连通分支 X 的开集的连通分支是开集。	components_are_ open_in_locally_ connected	forall X Tx:set, locally_connecte」 d_space X Tx -> forall C:set, C :e components_of X Tx -> open_in X Tx C
引理 26.1	引理 26.1。 X的子空间 Y 是紧致的，当且仅当每个开覆盖 X 中由开集合构成的 Y 有一个有限子覆盖。	compact_space_ subcover_ property	forall X Tx Y:set, topology_on X Tx -> Y c= X -> (compact_space Y (subspace_topology X Tx Y) <-> (」 forall Cfam:set, (forall U:set, U :e Cfam -> U :e Tx) -> Y c= Union Cfam -> exists F:set, F c= Cfam /\ finite_set F /\ Y c= Union F))
引理 26.4	引理 26.4。如果 Y 是紧致豪斯多夫 X 的子空间，并且 $x_0 \notin Y$，那么 x_0 和 Y 有不相交邻域。	Hausdorff_ separate_ point_compact_ set	forall X Tx Y x0:set, Hausdorf」 f_space X Tx -> compact_space Y (subspace_topology X Tx Y) -> x0 :e X -> x0 :e: Y -> exists U V:s」 et, U :e Tx /\ V :e Tx /\ x0 :e U /\ Y c= V /\ U :/\: V = Empty
引理 27.5	引理 27.5（勒贝格数引理）。每个开覆盖紧致度量空间都有一个勒贝格数。	Lebesgue_ number_lemma	forall X d:set, metric_space X d -> compact_space X (metric_topology X d) -> forall Cfam:set, open_cov」 er X (metric_topology X d) Cfam -> exists e:set, Lebesgue_number X d Cfam e

XX:24 13万行形式拓扑，两周

No.	topology.tex中的陈述	正式名称	正式陈述（统计信息，精确）
定理 27.6	定理 27.6。一个连续函数-从紧空间到度量空间的映射是均匀连续的。	uniform_continuity_on_compact	forall X Tx Y d f:set, compact_space X Tx -> metric_space Y d -> continuous_map X Tx Y (metric_topology Y d) f -> uniformly_continuous X Y d f
定理 29.1	定理 29.1。 一个局部紧豪斯多夫空间具有一个单点压缩。	one_point_compactification_exists	forall X Tx:set, locally_compact_Hausdorff X Tx -> exists Y Typ:set, one_point_compactification_of X Tx Y Ty p
定理 30.2	定理 30.2。 子空间与第一可数空间的可数乘积是第一可数的。 第一可数的。	countability_axioms_subspace_product	forall X Tx:set, first_countable_space X Tx -> (forall Y:set, Y c= X -> first_countable_space Y (subspace_topology X Tx Y)) /\ (forall Fam:set, (forall i:set, i :e index_set Fam -> first_countable_space (Fam[i]) (space_family_topology Fam[i])) -> first_countable_space (product_space Fam) (product_topology_full Fam))
引理 31.1	引理 31.1。 在一个 T_1 空间中，单点集是闭集。	T1_space_one_point_sets_closed	forall X Tx:set, T1_space X Tx -> forall x:set, x :e X -> closed_in X Tx {x}
定理 48.1	定理 48.1。空间 X 是贝尔空间当且仅当每个可数的交稠密开集的交集都是稠密的。	Baire_space_dense_Gdelta	forall X Tx:set, Baire_space X Tx <-> (forall Ufam:set, countable_set Ufam -> (forall U:set, U :e Ufam -> open_in X Tx U /\ dense_in X Tx U) -> dense_in X Tx (Inter Ufam))