并行计算

**大纲概要：**

本课程介绍现今最先进的超级计算机硬件和软件并行技术，内容上包括并行机体系结构、并行数值计算、并行算法设计和并行编程等。另外指导学生上机实验，从学习使用超级计算机到在Linux环境下独立安装、编写和调试并行程序。

**先修课程：**建议 计算机组成与结构 操作系统

**后继课程：**

**学习建议：**

1. 理解基础概念：

掌握并行计算的基本理论，如并行计算模型（如PRAM模型）、Amdahl定律、Gustafson定律等。熟悉并行机体系结构，包括多处理器系统、分布式系统和超大规模并行计算系统的特点。

2. 学习并行编程语言和工具：

熟悉常用的并行编程语言和库，如MPI（Message Passing Interface）、OpenMP、CUDA（用于GPU编程）等。掌握在Linux环境下的编程技巧和调试工具，如gdb、Valgrind等。

3. 动手实践：

多参与实验，实际操作是理解并行计算的关键。通过实验了解并行程序的执行过程和性能瓶颈。尝试在不同的硬件平台上运行并行程序，理解不同体系结构对程序性能的影响。

4. 优化并行程序：

学习如何分析和优化并行程序的性能。了解负载均衡、通信开销、内存访问模式等因素对并行程序性能的影响。使用性能分析工具（如Profilers）进行性能调优，寻找并解决性能瓶颈。

**数值计算及其计算机实现**

**大纲概要：**

本课程是计算机科学和技术专业的专业选修课，旨在学习和掌握在计算机上解决工程问题的常用数值计算方法及相关基础理论知识。学生将通过学习现代科学与工程计算中的常用数值计算方法和基本数学理论，锻炼编程能力和实践能力，掌握数值方法的实际应用背景，提升综合运用数值计算解决实际问题的能力和创新能力，并训练对数值计算实验结果进行分析和解释。课程内容涵盖并行计算基础、并行编程（包括MPI、OpenMP、CUDA）、实践与实验操作，以及数值计算方法在工程问题中的实际应用和研究。通过课程学习，学生将为后续课程和科学研究打下坚实基础。

**先修课程：高等数学，线性代数，程序设计**

**后继课程：无**

**学习建议：**

1. 打好基础：

在开始课程前，确保你已掌握基本的计算机科学概念、数据结构、算法和编程语言（如C/C++或Python）。了解一些基础的线性代数和微积分知识，这将有助于理解数值计算方法。

2. 积极参与课程活动：

认真听讲，积极参与课堂讨论，提出问题，和同学们分享自己的见解。完成所有实验和作业，这些实践机会对理解并行计算的概念和方法至关重要。

3. 动手实践：

多动手编写并行程序，使用MPI、OpenMP、CUDA等工具进行实际编程。在Linux环境下进行实验，熟悉各种调试和性能优化工具，如gdb、Valgrind等。

4. 注重实验结果分析：

不仅要完成实验，还要深入分析实验结果，理解并行算法的性能瓶颈和优化方法。学会使用性能分析工具，识别并解决程序中的性能问题。

5. 利用资源：

利用课程提供的教材和参考资料，阅读相关的学术论文和书籍，扩展自己的知识面。参与在线课程和研讨会，关注并行计算领域的最新研究和技术动态。

**自然语言处理导论**

**大纲概要：**

本课程主要讲述自然语言处理领域的基础知识、相关算法和最新研究进展，包括词法分析（英语单词识别，汉语分词，词性标注），句法结构分析，语义分析，篇章分析、问答系统、机器翻译等相关技术。同时，通过课程学习，使学生掌握自然语言处理的基本理论和常用工具及算法。通过本课程学习，使学生具备以计算机为背景思考语言学问题，对语言知识进行形式化描述的能力，了解计算语言学的基本理论和自然语言处理中的常用模型及算法，初步具备从事相关领域研究和开发工作的能力以及初步具备从事自然语言处理相关领域研究和开发的能力。

**先修课程：**建议 人工智能 概率论与数理统计

**后继课程：**无

**学习建议：**

1. 理解基础知识：

在开始课程前，确保你已掌握基本的计算机科学和编程知识，特别是Python编程。熟悉基本的语言学概念，这将有助于理解词法分析、句法结构分析和语义分析等内容。

2. 积极参与课程活动：

认真听讲，积极参与课堂讨论，提出问题，和同学们分享自己的见解。完成所有实验和作业，这些实践机会对理解自然语言处理的概念和方法至关重要。

3. 动手实践：

多动手编写NLP程序，使用常用的NLP库和工具，如NLTK、spaCy、gensim等。通过实际项目和实验，熟悉词法分析、句法分析、语义分析等技术。

4. 注重理论与实践结合：

不仅要理解理论知识，还要通过实践验证这些理论。在实验和项目中应用所学知识，分析和解决实际问题，理解NLP算法的实际应用效果。

5. 利用资源：

利用课程提供的教材和参考资料，阅读相关的学术论文和书籍，扩展自己的知识面。参与在线课程和研讨会，关注NLP领域的最新研究和技术动态。

**操作系统实践(进阶)**

**大纲概要：**

通过本课程的学习和上机实习，使学生掌握操作系统原理，操作系统设计基本思想。培养学生程序设计的能力和系统开发的能力。为以后学习系统设计开发相关课程及毕业后从事软硬件开发工作打下基础。本课程是一门实践性较强的课程，特别注重学生程序设计上机能力的培养。

**先修课程：操作系统**

**后继课程：**无

**学习建议：**

1. 准备工作：

确保在开始课程前已经掌握了操作系统的基础知识，尤其是先修课程《操作系统》中的内容。熟悉Linux操作系统的基本使用和命令行环境对于后续的实验和编程至关重要。

2. 积极参与实验和项目：

本课程侧重于实践能力的培养，特别是在操作系统的核心算法和系统调用的实现上。完成所有的实验和上机作业，通过动手实践来加深对课程内容的理解和掌握。

3. 深入理解Linux环境：

学习并掌握Linux系统的基本设计理念和主要特性。从安装和运行到常见的系统调用和调试工具（如gcc、gdb、makefile等），这些都是理解和开发操作系统实践所必需的技能。

4. 掌握xv6系统：

熟悉xv6操作系统的基本概念和架构，能够搭建实验环境并理解其系统调用过程和调度算法。通过编写和调试系统调用和调度器算法，加深对操作系统内部工作原理的理解。

5. 多维度学习：

不仅仅局限于课堂内容，还应阅读推荐的教材和相关文献，了解操作系统领域的最新进展和实践经验。参与课堂讨论和同学间的知识交流，能够拓展思维，加深对课程内容的理解。

6. 注重工程能力和文献阅读能力：

课程强调实践能力和工程应用，因此在实验和项目中，不仅要实现功能，还要能够分析和优化程序的性能和稳定性。同时，培养良好的文献阅读能力，及时了解并应用最新的研究成果和技术。

**现代CAD技术(A)**

**大纲概要：**本课程介绍现代CAD技术的基本概念与研究内容，讨论现代CAD技术中一些常见的关键性技术，并着重讲解CAD系统的各种开发技术，以及CAx系统（包括CAD、CAM、CAPP等）和相关设计使能工具DFx（DFP、DFA、DFM、DFT、DFQ等）的集成技术。本课程是计算机应用技术方面的选修课程，通过本课程的教学，使学生了解现代CAD技术的概念及研究方向，掌握CAD系统的开发和集成技术，具备独立完成简单CAD系统开发的能力。

**先修课程：**相关程序设计语言 C++ Python等

**后继课程：**无

**学习建议：**

1. 巩固编程基础，重点学习C++、Python和VBA。
2. 熟练掌握SolidWorks和AutoCAD软件，从基本建模开始逐步深入。
3. 学习DFP、DFA、DFM、DFT、DFQ等设计使能工具的概念和应用。
4. 选择一个小项目进行实践，通过项目巩固所学知识。
5. 理解现代CAD技术的基本概念、研究方向和关键性技术。
6. 积极参与课程实验和作业，特别是SolidWorks和AutoCAD的任务。

**计算机视觉**

**大纲概要：**

通过本课程的理论知识和实践环节的教学，使学生了解计算机视觉的基本原理和典型技术，并能据此解决计算机视觉应用中的一些具体问题，具体的理论教学内容包括两个大部分：传统的计算机视觉方法和智能计算机视觉技术。传统计算机视觉包括：图像处理相关基础、图像的特征、图像的对齐、二值分析、纹理特征及分析、图像分割、摄像机模型与多视几何、3D重建、光流计算、运动跟踪和目标识别等。智能计算机视觉技术包括：深度学习基础介绍、智能图像语义分割、智能目标识别技术、智能摄像机参数恢复与场景结构等。

本课程教学是一个兼顾理论算法和实践技能培养的教学环节，除了算法理论的教学环节，也特别注重学生实践技能的培养和训练。通过本课程的教学，培养学生分析问题、解决问题的能力；同时，通过实践教学的环节，培养学生计算机视觉算法设计的实践能力和计算机视觉软件开发的能力，使学生能将计算机视觉的算法和理论基础作为基本工具，在实际应用中解决实际问题，本课程教学为人工智能等后继课程的学习以及毕业后从事软件开发工作打下基础。

**先修课程：**数字图像处理

**后继课程：**无

**学习建议：**

1. 巩固数学基础：

复习线性代数、概率论与统计、微积分等数学知识，这些是理解计算机视觉算法的重要基础。

2. 掌握编程技能：

熟悉Python和C++编程语言，这两种语言在计算机视觉领域应用广泛。学习使用OpenCV、TensorFlow、PyTorch等计算机视觉和深度学习框架。

3. 理解传统计算机视觉方法：

深入学习图像处理相关基础、图像的特征提取与对齐、二值分析、纹理特征及分析、图像分割、摄像机模型与多视几何、3D重建、光流计算、运动跟踪和目标识别等内容。通过实际案例和项目加深理解，如实现简单的图像处理算法、特征提取和目标识别等。

4. 学习智能计算机视觉技术：

掌握深度学习基础知识，包括神经网络、卷积神经网络（CNN）等。进行智能图像语义分割、智能目标识别技术、智能摄像机参数恢复与场景结构等实践项目。参加在线课程或观看相关视频教程，深入理解智能计算机视觉技术的实际应用。

5. 实践训练：

积极参与课程实验和项目，动手实现算法，提升编程和调试能力。选择一个实际问题进行项目开发，例如开发一个简单的目标识别系统或3D重建项目，增强实践经验。

**游戏项目实践**

**大纲概要：**

本课程是面向计算机科学与技术的专业选修课程。本课程在介绍基本的游戏开发知识的基础上，将以游戏引擎为中心点，针对不同的游戏类型，阐 述不同游戏模块的设计策略和算法实现，并以项目实践贯穿整个课程，全面提高学生软件开发素质。本课程的内容包括：游戏开发简介；游戏编程基 础；游戏美术基础；游戏引擎简介，包括 UNITY 及 UNREAL；3D 游戏环境构建；光照系统；游戏角色动画；游戏 UI；游戏脚本技术；粒子系统；物理系 统；游戏开发插件；游戏软件架构；游戏性能分析；移动端游戏开发； VR/AR 游戏技术；游戏中的人工智能。本课程是一门实践性较强的课程，特别注 魏同权 重学生编程能力、项目设计实践能力的培养。

**先修课程：**数据结构、面向对象程序设计

**后继课程：**无

**学习建议：**

1. 强化编程基础：确保熟练掌握C++或C#，因为这两种语言在游戏开发中应用广泛，尤其是Unity主要使用C#，而Unreal引擎则主要使用C++。通过编写小程序或完成在线编程练习，提升自己的编程能力。熟悉面向对象编程和常用的数据结构、算法，有助于解决游戏开发中的复杂问题。
2. 熟悉游戏引擎：下载并安装Unity和Unreal引擎，通过官方教程或社区资源学习如何使用它们的基本操作和界面。重点掌握场景创建、对象管理、材质和光照系统的设置。通过创建简单的游戏场景，熟悉引擎的各项功能和工具，理解引擎的工作流程。
3. 项目实践：选择一个简单的游戏项目，从零开始设计、开发、测试和发布。这个过程中，可以涵盖角色动画、物理系统、粒子系统等课程内容。通过不断迭代和完善项目，实践课程中学习到的知识，提高解决实际问题的能力。同时，记录开发过程中的问题和解决方案，作为学习总结。
4. 关注前沿技术：了解虚拟现实（VR）和增强现实（AR）的基本概念和应用。使用相关的SDK（如Unity的XR插件和Unreal的ARKit）进行实践，尝试开发简单的VR/AR应用。关注行业动态，了解最新技术的发展趋势和应用案例，拓宽视野，为未来的发展做好准备。
5. 参与社区：加入游戏开发者社区，与其他学习者和开发者交流经验，参加游戏开发比赛或黑客松，提升实际开发能力。通过参与社区讨论、分享开发心得，获取更多的学习资源和支持。同时，关注行业大咖的博客、演讲，学习他们的开发经验和技巧。

**数据挖掘**

**大纲概要：**

本课程是计算机科学与技术专业的专业选修课，是一门面向数据挖掘和数据分析的理论研究和实践应用型课程。本课程围绕数据挖掘和知识发现的概念、原理、算法等理论知识，并结合前沿的多领域的实践应用研究展开，是以前高等数学和基础专业课程在高级应用领域的扩展延伸，是人工智能领域的重要基础课程。通过学习使学生掌握关于数据挖掘领域的基本概念，技术方法，基本原理，并了解在当前实际的研究热点中这些知识的应用；通过课程指导掌握获取本领域新知识的方法；通过应用实践，掌握多领域多样化问题的分析和解决技术；课程的重点是深度掌握这些基本理论知识的基础上，学会分析问题、理解各种数据挖掘算法的原理、优缺点和改进策略，并初步运用这些理论知识和算法技术解决多领域应用实践中的数据分析和数据挖掘问题，为学生在今后的工作和进一步的科研学习建立知识的桥梁。

**先修课程：**数学分析、线性代数、概率论与数理统计、C语言、面向对象程序设计

**后继课程：**无

**学习建议：**

1. 掌握基本概念和理论：系统学习数据挖掘和知识发现的基本概念和原理，理解常见的数据挖掘算法，如分类、聚类、关联规则挖掘、回归分析等。确保对每种算法的理论基础、适用场景、优缺点和改进策略有深刻理解。这些基础知识是后续学习和实践的关键。

2. 熟悉数据挖掘工具和编程语言：熟练使用数据挖掘常用的编程语言，如Python和R。学习使用常用的数据挖掘和机器学习库，如Scikit-learn、TensorFlow、Keras等。通过实践项目，掌握数据预处理、特征工程、模型训练与评估的基本流程和技巧。

3. 注重实践应用：通过课程指导和实践项目，掌握在多领域中分析和解决数据挖掘问题的方法。选择一些实际案例进行实践，如电商推荐系统、社交网络分析、医疗数据分析等。通过项目实践，提升分析问题、设计解决方案、实施算法和评估结果的能力。

4. 关注前沿研究和应用：定期阅读数据挖掘领域的最新研究论文和技术报告，了解当前的研究热点和发展趋势。参加学术会议、研讨会或线上讲座，与业内专家交流，拓宽视野。通过了解前沿研究，找到自己的兴趣点，并探索潜在的研究方向。

5. 培养问题解决能力：在学习理论和实践的过程中，注重培养分析问题、理解算法原理、优化算法性能的能力。多进行独立思考和实践，尝试不同的方法解决问题，并总结经验教训。通过不断尝试和优化，提升解决复杂数据挖掘问题的能力。

**多智能体系统与实践**

**大纲概要：**

随着人工智能技术的发展，我们所需要面对的通常是一个多智能体系统，通过有效技术手段实现多智能体系统种智能体之间的高效协作是未来人工智能发展的重点方向。本课程关注多智能体系统、多智能体协同、多智能体强化学习等一系列多智能研究热点。

**先修课程：**建议 人工智能 高等数学 线性代数 概率统计

**后继课程：**无

**学习建议：**

1. 系统学习基础理论：认真学习多智能体系统的基础理论，包括智能体建模、多智能体协同和对抗方法。这些基础知识是理解更高级概念和应用的前提。

2. 关注多智能体强化学习：深入学习多智能体强化学习的相关内容，包括通信学习和多智能体演员-评论家方法。这些技术在多智能体系统中应用广泛，理解其原理和应用场景有助于掌握前沿技术。

3. 理论结合实践：在学习理论知识的同时，通过实际案例和应用项目进行实践。通过实践掌握设计合理有效的多智能体系统算法，并能够在实际应用中解决问题。

4. 参与讨论和研究报告：积极参与课堂讨论和课后研究，撰写研究报告。通过研究报告，深入分析和总结多智能体系统的理论和应用，提高学术写作和研究能力。

5. 利用参考资料：阅读并研究《分布式人工智能》一书，了解更多关于多智能体系统的最新研究成果和应用案例。参考资料中的内容将有助于加深对课程内容的理解。

6. 课外拓展：关注人工智能领域的最新研究动态，阅读相关领域的学术论文和技术报告，参加相关的研讨会和讲座，了解多智能体系统的最新发展和应用前景。

**计算机网络工程**

**大纲概要：**

本课程是计算机科学与技术专业的专业选修课。通过本课程的学习，使学生掌握计算机网络工程的基本方法，了解计算机网络工程中涉及的关键技术和解决方法，掌握计算机网络工程从规划、选型、施工、测试到管理的全过程，掌握典型局域网、广域网、网络互联和接入技术，学会基本网络设备交换器、路由器和服务器等设备的工作原理与配置方法，为学生在今后的工作中能参与计算机网络工程打下良好基础。

**先修课程：**操作系统 计算机网络

**后继课程：**无

**学习建议：**

先修知识： 在开始修读《计算机网络工程》之前，学生应当具备扎实的计算机网络基础知识。这包括理解 OSI 模型和 TCP/IP 协议族的结构和功能，熟悉常见的网络设备如路由器和交换机的工作原理，以及基本的网络配置和管理技能。此外，对于网络安全和数据传输原理的基本理解也是必要的。

实验能力： 本课程强调理论与实践相结合。学生需要具备一定的实验能力，能够通过网络仿真软件如Cisco Packet Tracer或者使用实验室设备进行网络配置和仿真实验。这不仅有助于理解课堂上学到的理论知识，还能够培养学生的问题解决能力和实际操作技能。

编程基础： 虽然《计算机网络工程》课程通常不要求深入的编程技能，但是一些基本的编程能力可以帮助学生理解和实现网络协议和算法。例如，使用Python等语言编写简单的网络应用或者自动化脚本能够加深对网络工程原理的理解。

数学基础： 理解网络工程中的一些复杂算法和协议设计通常需要一定的数学基础。学生应当了解基本的离散数学概念，如图论和概率论，以及在网络领域中的应用。这些知识可以帮助学生更好地分析和优化网络性能，设计可靠的网络架构。

资源利用： 学生需要熟悉和善于利用网络工程领域常用的现代工具和仿真软件。这些工具包括但不限于网络仿真软件（如GNS3、Wireshark）、网络管理工具（如Nagios）和网络安全工具（如Snort）。通过实际操作这些工具，学生能够深入理解网络工程中的实际应用和挑战。

学习方法： 成功修读《计算机网络工程》的关键在于采用有效的学习方法。建议学生在课前预习相关章节，积极参与课堂讨论和实验，及时复习和总结课堂内容，并通过解决实际问题和项目来巩固学习成果。与同学和教师的积极互动也是提升学习效果的重要途径。

**网络安全基础**

**大纲概要：**

本课程是一门面向计算机学院计算机科学与技术专业高年级本科学生开设 的专业选修课，课程既是一门技术理论课，又是一门实践性极强的课程。课程理 论与实践紧密结合，应用性强。 课程主要要求学生从三个方面掌握网络安全的基本原理、主要技术以及解决 方案。 1．网络安全体系结构：介绍网络安全的研究体系、研究网络安全的意义、 评价网络安全的标准、以及网络安全协议、技术等。 2．网络安全主流技术：根据应用需求和安全策略，系统掌握并综合运用各 种网络安全技术，包括现代密码学及其应用技术、协议安全性分析、PKI/PMI、 VPN/IPSec、安全审计、防火墙技术、入侵检测技术、病毒/恶意代码及防范技术、 网络攻击及防范技术、无线及移动网络安全技术等。 3．网络安全工程：用系统工程的方法贯穿网络安全系统的设计、开发、部 署、运行、管理和评估

**先修课程：**计算机网络 操作系统

**后继课程：**无

**学习建议：**

1. 网络安全体系结构与原理：

理解网络安全的体系结构，包括其研究意义和评估标准。学习常见的网络安全协议和技术，如SSL/TLS、IPSec等，以及它们在网络安全中的应用。

2. 现代密码学与应用：

深入学习现代密码学的基本原理，包括对称加密、非对称加密和哈希算法的理解。理解密码学在安全通信和数据保护中的关键作用。

3. 网络安全技术与工具：

掌握常见的网络安全技术，如防火墙、入侵检测系统（IDS/IPS）、虚拟私人网络（VPN）等的工作原理和应用场景。学习安全审计和漏洞评估方法，以及如何利用这些技术保护网络系统免受攻击。

4. 网络安全工程与实践：

学习使用系统工程方法来设计、开发、部署和管理网络安全系统。理解安全策略的制定和实施，以及安全事件响应和管理的基本流程。

5. 实验与实际操作：

参与实验课程，通过实际操作加深对理论的理解。运用学到的知识解决实际的网络安全问题和挑战。

**存储技术基础**

**大纲概要：**

应用中需要被存储和管理的数据量急剧膨胀，存储相关岗位出现了巨大缺口。《存储技术》是面向计算机科学与技术专业高年级本科生的专业选修课程，能帮助学生了解主流的企业级存储介质、接口、架构及代表性解决方案，掌握存储产品的部署、配置、运维等技能，为将来走向工作岗位或从事相关研究打下坚实基础。

**先修课程：**计算机网络

**后继课程：**无

**学习建议：**

1. 预习相关背景知识： 理解计算机网络和数据存储的基础概念对于后续深入学习存储技术至关重要。建议提前学习计算机网络的基本原理，包括网络体系结构、传输协议（如TCP/IP）、网络安全和数据传输机制等。此外，熟悉数据存储的基础概念，如数据的组织结构（文件系统、数据库）、数据存储层次结构（主存、磁盘、固态硬盘）以及数据的读写操作原理。

2. 参与实验和实践： 虽然课程本身可能没有专门的实验和实践学时，但建议自主进行相关实验以加深理解。可以使用虚拟仿真实验平台（如VirtualBox、VMware等），搭建存储设备的虚拟环境，模拟不同存储设备的工作原理和操作。通过实际操作，可以更直观地理解存储技术的工作方式和性能特征。

3. 提前准备和复习： 由于《存储技术》课程涵盖的内容广泛且深度较大，建议提前规划学习时间。合理安排复习和课后作业的时间，确保对每周的课程内容都有充分的理解和掌握。可以通过预习课程大纲、查阅相关教材和文献，准备好课前可能需要的背景知识，以便能够更快速地跟上课程进度并深入理解讲授的内容。

**模式识别与机器学习**

**大纲概要：**

本课程是计算机科学与技术专业本科生（大三）的一门专业选修 课。讲授模式识别与机器学习领域中的基本概念、典型算法，及其在 计算机视觉、自然语言处理中的应用，并初步培养学生的科研兴趣

**先修课程：**数学分析 线性代数 概率论与数理统计

**后继课程：**无

**学习建议：**

1. 预备知识准备： 在课程开始前，建议复习和加强以下基础知识：确保对高等数学、线性代数和概率论有扎实的理解，因为这些知识在课程中会频繁应用，特别是在算法推导和理论分析中。编程能力： 课程涉及算法实现和数据处理，熟练使用Python或者其他数据科学工具（如NumPy、SciPy）将对理解和实验有帮助。

2. 理论与实践结合： 虽然课程中没有实验学时，但建议通过自主学习和实验平台进行相关实践，尤其是对课程中提到的算法进行实现和调试。这有助于加深对模型工作原理的理解，提高编程能力和算法应用能力。

 深入学习重点内容：贝叶斯决策理论和概率图模型： 这些是模式识别和机器学习的基础，理解它们对后续学习和应用至关重要。支持向量机和深度学习： 这些是当前热门的机器学习技术，对它们的理解和实现能力将极大地增强你在相关领域的竞争力。

3. 作业和大作业的重视： 课程中的作业和大作业占据了相当比例的评分，建议及时完成作业并且注重质量，这不仅有助于理解和巩固知识，也是提升平时成绩的关键。

服务器维护及网站建设 没找到这门课的开课记录