# 项目：缓存一致性协议建模与验证

本项目涉及基于SPIN(spinroot.com)平台的建模、仿真与模型检查，项目的主要内容是缓存一致性协议的建模与验证。模型检查工具是xspin（现在在SPIN官网中叫ispin），需要熟悉这个工具的使用。

## 建模

建模语言是Promela，该模型由异步执行、通过在通道上发送/接收消息进行通信的进程组成。（说明，在进行本次项目之前，需要仔细学习Promela建模语言，官网<http://spinroot.com/spin/Man> ）

## 仿真

可以对模型进行仿真或验证。在仿真模式中会生成单个执行，并使用消息序列图格式显示出来。消息序列图是统一建模语言UML中常用的一种图。

## 声明需求

可以在模型中插入assert语句来声明需求。如果assert失败，则会报错。你也可以使用LTL公式。SPIN会将这些公式编译成永不声明的公式(never claims)，这些公式正是教材5.2.1节中的监视器（可以将它们直接写为Promela模型）。

## 模型检查

SPIN中的基本验证算法是枚举深度优先搜索。SPIN中已经在该算法基础上施加了大量优化，从而使其能够处理复杂模型。验证器的性能严重依赖于参数的设置。

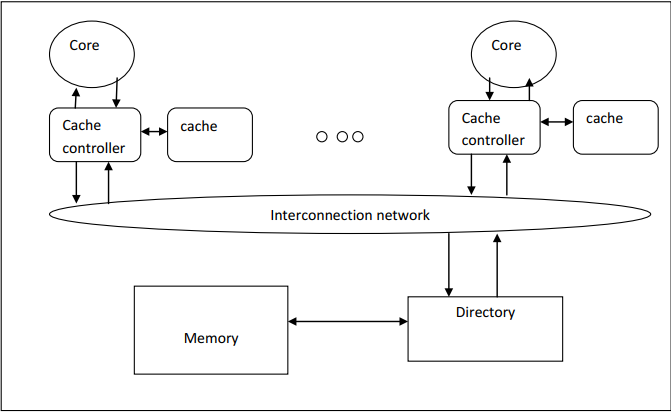
## 项目概述

学习完基本工具后，需要对缓存一致性协议进行建模和分析。

这里简单介绍一下缓存一致性协议。

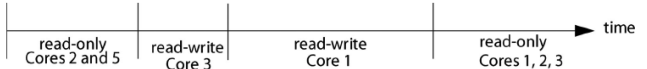
**缓存一致性协议**：现代计算机系统包含多核处理器。通常每个核(core)有自己的缓存，所有核可以在一块共享存储上进行读写操作。因为每个核可以更改自己缓存上的数据，不同缓存上同一数据的多个副本可能导致不一致(inconsistency)。例如，多个缓存访问同一数据，其中一个缓存更新了这个数据，然后其它缓存不会读到最新的数据。缓存一致性协议的目标就是确保多个缓存的数据副本保持最新。

本项目将实现一个简单的缓存一致性协议。如下图所示，系统包括一组核、对应的缓存、缓存控制器、一个页表(directory)及内存。



一致性的基础是SWMR不变式(single-writer–multiple-reader invariant)。对于任何给定的内存地址，在任何时间，只有一个核在进行写操作(可能包含读操作)或者一些核在进行读操作。因此，不存在一个时间，在该瞬间一个核写入某个内存位置，同时该位置有任何其他核在读取或写入。

在SWMR不变式的基础上，一致性要求给定内存地址的值是正确传播的(propagated correctly)。请看下图的例子



即使已经保证了SWMR不变式，如果在第一个只读时间段内，核2和5读到不同的值，系统也是不一致的。同样的，如果核1不能读到最新的核3在其读写时间段里写入的值，或者任何核1、2、3不能读到最新的核1在其读写时间段里的写入的值，系统也是不一致的。因此，一致性的定义需要在SWMR不变式的基础上扩展数据值不变式(data value invariant)，它是关于数据值如何从这一时间段传入下一时间段的约束。

一致性协议的目标是通过施加这些不变式来保持一致性。为了实现这些不变式，将每个存储结构(缓存和内存)与一个有限状态机相关联，这个有限状态机被称为一致性控制器(coherence controller)。这些控制器构成一个分布式系统，在这个系统里，控制器间相互交换信息，确保对于每个块(block)来说，SWMR和数据值不变式都得到保证。有限状态机间的交互由一致性协议决定。

本次项目的缓存一致性协议是directory协议的变体。在directory协议中，directory维护每个块的状态，缓存控制器发送请求给directory。directory要么响应这些请求，要么将这些请求转发给其它缓存控制器，然后响应。在大多数directory协议中，directory排序一致性事务(coherence transaction)。多个控制器可能同时向directory 发送请求，事务顺序由请求在directory中序列化的顺序决定.

我们将控制器的动作简化成一张表，行代表内存块的状态，列表示不同的事件。将表中的状态/事件条目称为迁移（transition），与块B相关的事件E的迁移包括（a）当E发生时采取的动作和（b）块B的下一状态。用“动作/下一状态”来表示这种转换，当下一状态和当前状态一致时，省略“下一状态”，表1和表2分别表示缓存控制器和directory的规约。表中空条目表示这种迁移不会发生。

表1：缓存控制器

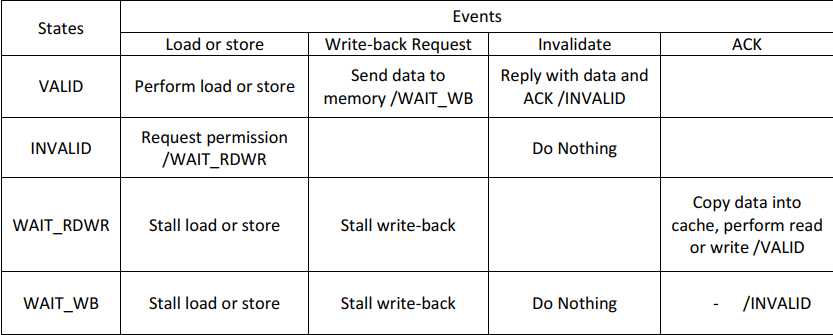
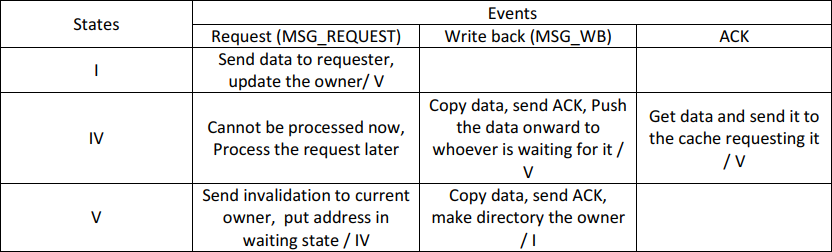


表2：directory



协议的详细解释说明：

在初始状态下，所有缓存块和内存块的状态是I(dle)/INVALID。这意味着内存块的拥有者是directory，所有核没有权限进行读写操作。

对于核或者directory的每个动作，协议定义如下：

**(针对读写操作，此时内存状态是VALID)**

如果核想要读写的内存地址的状态是VALID，说明缓存已经获得该块的权限，可以执行相关操作。如果块的状态是INVALID，缓存控制器向directory发送针对该块的请求(MSG\_REQUEST)，并把自己的状态设为WAIT\_RDWR。

directory会收到来自缓存控制器的请求。如果没有核拥有被请求块的权限，并且没有缓存在等待这个内存块的数据(如状态为I)，directory会把数据发送给请求的缓存，更新该内存块的信息(更新块的拥有者，设置状态为V)。需要注意的是directory逐个处理请求，如果directory在另一个缓存等待特定块的时候收到请求，它不会处理这个请求。

如果内存地址(内存块)状态是V，意味着另一缓存拥有这个块，directory会给当前权限拥有者发送invalidation消息(MSG\_INVALIDATE)，在获得当前块的数据并且收到当前权限拥有者的ACK消息(MSG\_ACK)之前directory不会将权限给发送请求的缓存。在等待ACK消息的过程中，directory 将块的状态设为IV(等待状态)。在这个时间段里，任何针对这个块的请求都会被延迟处理。

**(针对读写操作，此时内存状态是WAIT\_RDWR)**

如果缓存控制器是在IV状态下，有load或store请求，因为有其它请求正在处理过程中，控制器需要延迟处理这些请求。

**(核的write-back请求)**

如果核想要释放缓存空间，将一个块放回内存中以替换另一个地址的块，缓存控制器向directory发送所需内存地址的写回请求(MSG\_WB)，向directory发送数据，并将状态设为WAIT\_WB。

一旦收到这个请求，directory将把数据复制到内存中，向控制器发送ACK消息。控制器收到ACK消息后将状态设为INVAILD。

如果另一个缓存在等待这个块(状态IV)，数据将会被转发给等到的缓存。状态改为V，拥有者的信息更新(例如，正在等待的缓存)。否则，将块的状态设为I。

**(来自directory的invalidate事件)**

当控制器收到来自directory的invalidate消息，这意味着另一个缓存控制器正在请求内存中相应的块，控制器将数据和ACK消息发送给directory。然后将自己的状态设为INVALID。

(协议剩下的情况在表中)

**备注：**

在本次项目中，我们假定缓存控制器通过独立的请求通道(individual request channels)发送请求，通过独立的回复通道(individual reply channels)接收回复消息。例如，每个缓存控制器通过两个通道与directory进行通信，一个用来发送请求和ACK消息给directory，另一个用来接收来自directory的消息。

缓存控制器在WAIT\_\*状态下应该推迟处理写回(write-back)请求和读写请求，直到控制器能够处理为止，在本次项目中，控制器忽略这些请求。

为了降低项目的复杂性，在项目过程中可以假定所有的缓存和内存是一样的，只有一个bit，这样就不需要地址映射。

## 项目内容

1. 项目模型框架已经给出，请根据上述缓存一致性协议模型将代码补充完整(可以根据需要修改框架代码)，最后应该实现上述协议。实现的模型应该将缓存数量作为参数传入(不能将缓存固定在某一数量)。需要考虑的是：设置多大的缓冲区才能满足模型节点间通信的需要，以及该如何维护缓冲区。
2. 在特定配置下，对模型进行仿真，观察实验现象(例如，当有两个核和三个核时，尝试观察不同的情况，例如当两个缓存控制器发送对另一个缓存控制器所持有的同一个块的请求时)。多次重复实验时应该改变消息在各种队列中累积的顺序并进行处理，但不要违反协议不变式要求。
3. 实验提供监控程序monitorSWMR，可以用来确保在任何时候拥有读写权限的核的数量不多于1个。在实验中应该使用assertion来确保其它的安全属性。运行验证程序以检查模型是否满足上述第(2)部分中选择的初始配置的这些要求。
4. 设置实验以检查模型的可达状态数、验证程序使用的时间和内存如何随缓存(核心)数而增加。具体做法，将缓存数量从2增加到6，将上面的数字画图展现。

## 实验环境及参考资料

实验自行需要配置spin运行环境，学习Promela建模语言和ispin工具。

1. [spin官网](<http://spinroot.com> )官网有详细资料。
2. [spin/ispin安装教程](<http://spinroot.com/spin/Man/README.html> )，建议使用Linux环境。
3. Promela建模语言参考资料：
   1. Promela官方手册: <http://spinroot.com/spin/Man/promela.html，http://spinroot.com/spin/Man/Intro.html>
   2. overview of Promela : 简单介绍Promela的资料，英文版（随实验指导书一块给出）。
   3. 自动模型检测-SPIN的建模语言Promela.pdf，一份中文版的资料。 <https://doc.mbalib.com/view/45c12466d3ce1bc121da49ff133827ef.html>
   4. 实验说明英文版：这个文件是英文版的实验说明，可供参考。项目验收要求以指导书为准。

## 代码说明

主要提供两个文件：

1. spin-template:缓存一致性协议的框架代码(Promela版)，需要根据协议将缺少的内容补充完整。
2. TrainController2:(Promela版)：一个简单的控制器程序，可参考其中Promela语法等。

备注：在spin-template中有注释说明哪些情况可以不做处理。

## 项目验收

本次项目/实验具有一定的难度，分组完成，每组2-3人。小组成员应该加强研讨，共同完成。

本项目的验收分为两部分，一是项目答辩，二是实验报告与代码提交。

项目答辩为汇报形式，需要准备PPT。汇报内容需覆盖主要实验内容，至少包含三部分：

* 对协议的理解。
* 协议的实现。实现思路，缓存区的设置与管理等。
* 实验现象观察。增加缓存（核心）数量，观察模型可到达的状态、程序使用时间和内存的变化。

实验报告与代码提交的内容包括：(1) Promela模型代码，对应项目内容1；(2) 解释分析仿真的设置和结果，对应项目内容2；(3) 安全属性和验证结果，对应项目内容3；(4) 实验内容4的图表。

验收时间：初定于2019年1月7日，具体时间地点另行通知。