CoCoA：大规模机器学习的分布式优化通用框架

去年，Michael I. Jordan 实验室发表论文《CoCoA: A General Framework for Communication-Efficient Distributed Optimization》提出了一种用于机器学习的分布式优化的通用框架 CoCoA。机器之心技术顾问 Yanchen Wang 对该研究进行了深度解读。

**引言**

在做深度学习时，现代数据集的规模必需高效的设计和开发，而且理论上算法也要进行分布式优化。分布式系统可以实现可扩展性——不管是垂直扩展还是水平扩展，提升计算和存储能力；但同时也让算法设计者面临着一些独特的难题。其中一个尤其关键的难题是在机器学习负载的背景下，开发能有效地协调机器之间的通信的方法。对大多数生产集群而言，网络通信确实比单台工作机器上的本地内存存取要慢得多；但是，扩展单台机器显然不可行。这个问题还可以更加复杂，本地计算和远程通信之间的最优平衡取决于数据集的特定属性（比如维度、数据点的数量、稀疏度、偏度等）、分布式系统的特定属性（比如数据存储格式、分布式方案和数据存取模式等逻辑方面的设计，以及网络层次结构、带宽、计算实例规范等物理方面的条件）和负载的特定属性（比如简单的 ETL 过程肯定不同于 logistic 回归的迭代式拟合）。因此，算法设计者必须要让他们的优化/机器学习算法具有足够的灵活性，从而在保证快速收敛的前提下实现特定分布式系统的「计算-通信」的最优平衡。

CoCoA 是加州大学伯克利分校 Michael I. Jordan 的实验室最近提出的一个框架，通过对多种多样的优化问题的智能分解而实现了上述目标。通过自由选择原始或对偶的目标来解决，该框架成功利用了凸对偶性（convex duality），从而可将全局问题分解成一揽子可在工作机器上有效并行求解的子问题，并且可以将局部更新组合起来以一种可证明的方式确保快速全局收敛。CoCoA 有两个显著优势：1）在每台工作机器上都可以最有效地运行任意本地求解器；2）计算-通信的权衡可以作为形式化问题的一部分进行调节，因此可以对每个不同的问题和数据集进行有效的调节。

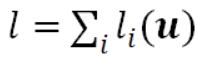
根据数据在分布式集群上的分布情况（不管是根据特征还是根据数据点），CoCoA 可以将全局问题分解成近似的局部子问题，推荐应求解原始目标或是对偶目标。每个子问题都使用当前最佳的现成单台机器求解器解决，然后在单一一步 REDUCE 步骤中将来自每次迭代的局部更新组合起来（REDUCE 这个术语借用自 MAP-REDUCE）。实验表明 CoCoA 可以在 SVM、线性/logistic 回归和 lasso 算法上实现最高 50 倍的加速。

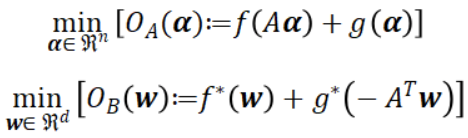
在这篇报告中，我们将了解 CoCoA 的核心思想和最重要的结论，感兴趣的读者可以在参考文献中找到详细论证和更多实验。本报告的目标是启发我们分布式机器学习领域的读者以及邀请更多人加入到我们的讨论中，与我们交流知识以及为我们的技术社区做出贡献。

**问题设置**

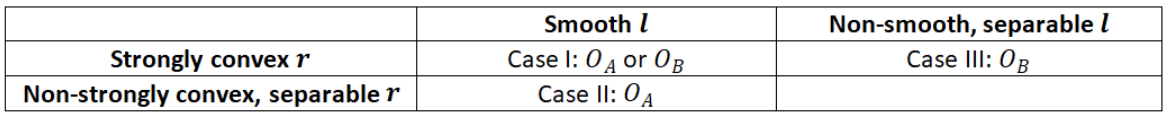
CoCoA 的目标是解决机器学习算法中普遍存在的下面一类优化问题：

https://image.jiqizhixin.com/uploads/wangeditor/d02dc172-e784-4a64-a74e-e35943bf6bfa/40842WX20170827-104916@2x.png

其中 l 和 r 是向量变量 u 的凸函数。在机器学习领域，l 通常是一个单独的函数，表示所有数据点的经验损失（empirical loss）的总和；而https://image.jiqizhixin.com/uploads/wangeditor/d02dc172-e784-4a64-a74e-e35943bf6bfa/32453WX20170827-105201@2x.png，表示 p 范数的正则化项。SVM、线性/logistic 回归、lasso 和稀疏 logistic 回归都属于这个类别。这个问题通常是在原始空间或对偶空间解决的。在我们的讨论中，我们将这种原始/对偶问题抽象成了下面的 Fenchel-Rockafeller 对偶形式：



其中 α 和 w 是原始/对偶变量，A 是包含数据点列向量的数据矩阵，而 f\* 和 g\* 则是 f 和 g 的凸共轭。非负的对偶间隙（duality gap）https://image.jiqizhixin.com/uploads/wangeditor/d02dc172-e784-4a64-a74e-e35943bf6bfa/18931WX20170827-105310@2x.png，其中 w(α)=∇f(Aα)，为原始或对偶的次优性（suboptimality）提供了一个可计算的上限，并且可以在强凸性下在最优解点减少到零。它可以用于验证解的质量和用作是否收敛的标志。根据 l 的平滑度和 r 的强凸性，我们可以将目标 l(u)+r(u) 映射到 OA 或 OB：

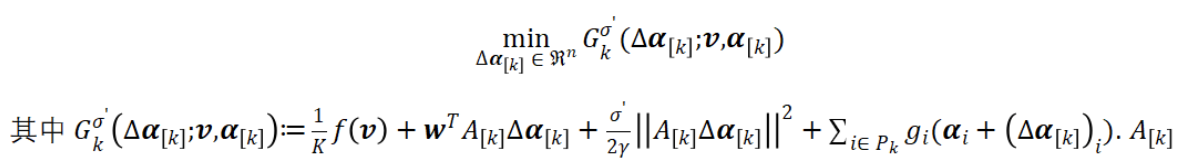


每种情况的典型案例有：弹性网络回归是 Case I，lasso 是 Case II，SVM 是 Case III。这里省略了推倒过程。

**CoCoA 框架**

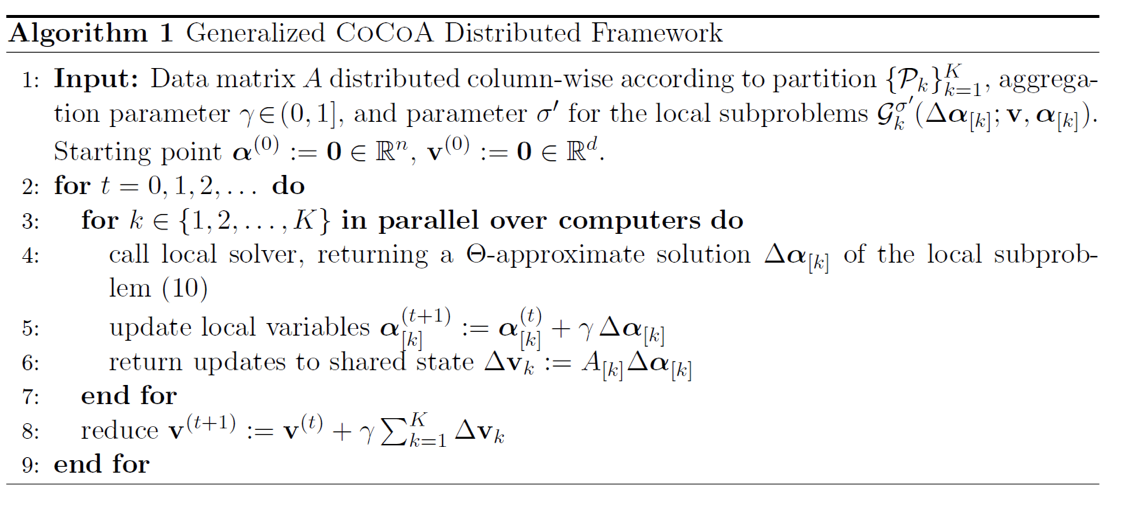
要在数据分布在 K 台机器上时最小化目标 OA，我们需要将计算分配给 K 个局部子样本并在每次全局迭代过程中将 K 个局部更新组合起来。首先将数据矩阵 A 的列分成 K 个数据分区https://image.jiqizhixin.com/uploads/wangeditor/d02dc172-e784-4a64-a74e-e35943bf6bfa/15702WX20170827-105458@2x.png。对于每个工作机器 k，定义https://image.jiqizhixin.com/uploads/wangeditor/d02dc172-e784-4a64-a74e-e35943bf6bfa/48227WX20170827-105537@2x.png，其中当 i∈Pk 时，https://image.jiqizhixin.com/uploads/wangeditor/d02dc172-e784-4a64-a74e-e35943bf6bfa/84325WX20170827-105613@2x.png，否则https://image.jiqizhixin.com/uploads/wangeditor/d02dc172-e784-4a64-a74e-e35943bf6bfa/11292WX20170827-105643@2x.png。注意这种表示方式与数据的分布方式无关——数据矩阵https://image.jiqizhixin.com/uploads/wangeditor/d02dc172-e784-4a64-a74e-e35943bf6bfa/58524WX20170827-105712@2x.png的维度 n 和 d 各自都可以表示特征的数量或数据点的数量。这种可互换性是 CoCoA 的一个显著优势——提供了灵活的数据分区方式，通过特征或数据点都可以，这取决于哪个更大以及使用了哪种算法。

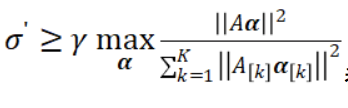
分布 g(α) 很简单，因为它是可分的https://image.jiqizhixin.com/uploads/wangeditor/d02dc172-e784-4a64-a74e-e35943bf6bfa/88026WX20170827-105800@2x.png ；但要分布 f(Aα)，我们需要最小化它的二次近似。我们定义了以下仅读取局部数据子样本的局部二次子问题：



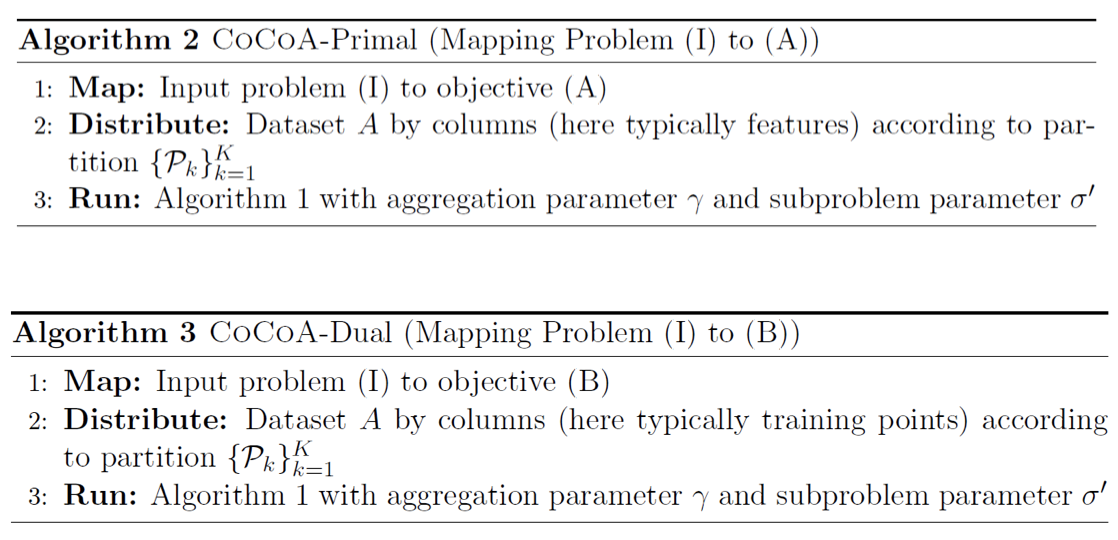
表示机器 k 上的一组列，类似于 https://image.jiqizhixin.com/uploads/wangeditor/d02dc172-e784-4a64-a74e-e35943bf6bfa/60481WX20170827-105910@2x.png是来自前一次迭代的共享向量，https://image.jiqizhixin.com/uploads/wangeditor/d02dc172-e784-4a64-a74e-e35943bf6bfa/90474WX20170827-105942@2x.png表示局部变量αi 在所有 i∈Pk 上的变化，而且在 i∉Pk 时为零。这个子问题是围绕固定的 v 的临近区域 f 的线性化，可以通过大多数有效的二次优化求解器解出。可以直观地看到，https://image.jiqizhixin.com/uploads/wangeditor/d02dc172-e784-4a64-a74e-e35943bf6bfa/28961WX20170827-110019@2x.png试图随局部https://image.jiqizhixin.com/uploads/wangeditor/d02dc172-e784-4a64-a74e-e35943bf6bfa/58882WX20170827-110047@2x.png变化而非常近地近似全局目标 OA。如果每个局部子问题都可以得到最优解，那么 REDUCE K 个更新可以被解读成一个独立于数据的、与数据块无关的近似 OA 的 f 部分的步骤。但是，和传统的近似方法不同，CoCoA 并不需要完美的局部解。相反，它容许局部次优性（定义为与最优的期望绝对偏差），并且会将其整合进自己的收敛边界中，下面就可以看到。有些实践者想复用已有的针对特定问题、数据集和机器配置优化的单个求解器，对于他们而言，这是一个巨大的优势。

完整算法如下：

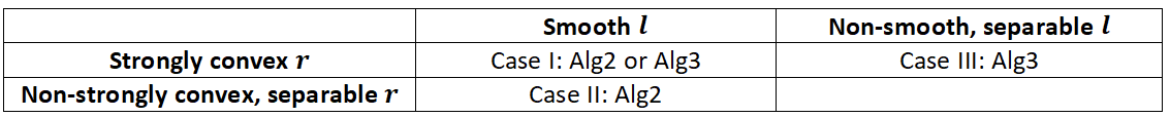


其中有两个可调节的超参数：γ 控制了工作机器的更新的组合方式，σ' 则表示数据分区的难度。实际上，对于一个给定的 γ∈(0,1]，我们设 σ':=γK。γ=1 且 σ'=K 能保证最快的收敛速度，尽管理论上任何都应该是足够的。详细证明请参阅原论文。

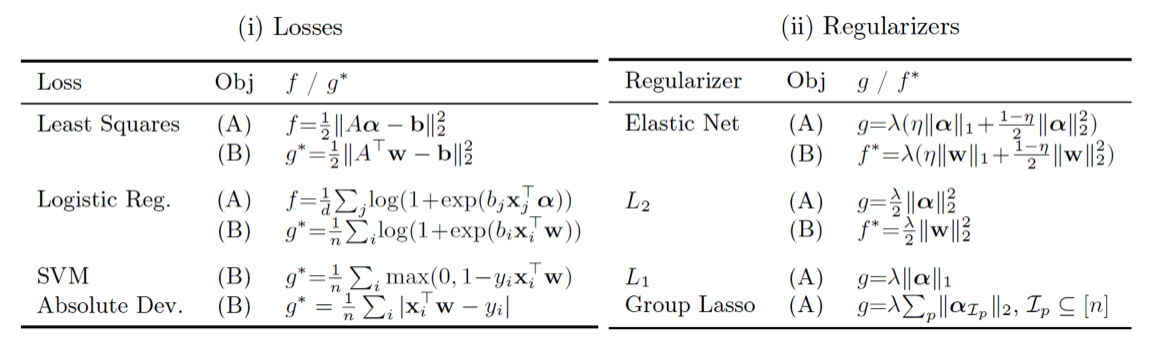
CoCoA 的原始-对偶灵活性是一大主要优势。尽管事实上我们一直都在求解 OA，但我们可以自由地把它看作是https://image.jiqizhixin.com/uploads/wangeditor/d02dc172-e784-4a64-a74e-e35943bf6bfa/82036WX20170827-110243@2x.png的原始或对偶——如果我们将这个原始问题映射成了 OA，那么 OA 就是原始；如果我们将其映射成了 OB，那么 OA 就是对偶。将 OA 看作原始让我们可以求解 lasso 这样的非强凸性正则化项，通常这是当数据按特征分布，而不是按数据点分布的时候。这能很好地适用于 lasso、稀疏 logistic 回归或其它类似 L1 的诱导稀疏性的先验（sparsity-inducing priors）。求解 CoCoA 的这种原始变体每次全局迭代的通信成本为 O(数据点的数量)。另一方面，将 OA 看作对偶让我们可以考虑非平滑损失，比如 SVM 的 hinge loss 或绝对偏差损失，而且当数据是按数据点而非特征分布时效果最好。这种变体每次全局迭代的通信成本为 O(特征数量)。下面是对这两种 CoCoA 变体的总结：

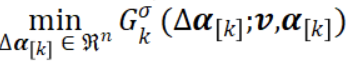


复用上面的表格，我们现在得到：



下表给出了在 CoCoA 框架中构建的常见模型的例子：



在原始的设置（算法 2）中，局部子问题变成了在局部数据块上的二次问题，其中仅有局部的https://image.jiqizhixin.com/uploads/wangeditor/d02dc172-e784-4a64-a74e-e35943bf6bfa/44284WX20170827-110531@2x.png是正则化的。在对偶的设置（算法 3）中，经验损失仅应用于局部https://image.jiqizhixin.com/uploads/wangeditor/d02dc172-e784-4a64-a74e-e35943bf6bfa/62236WX20170827-110531@2x.png，这使用了由一个二次项近似的正则化项。

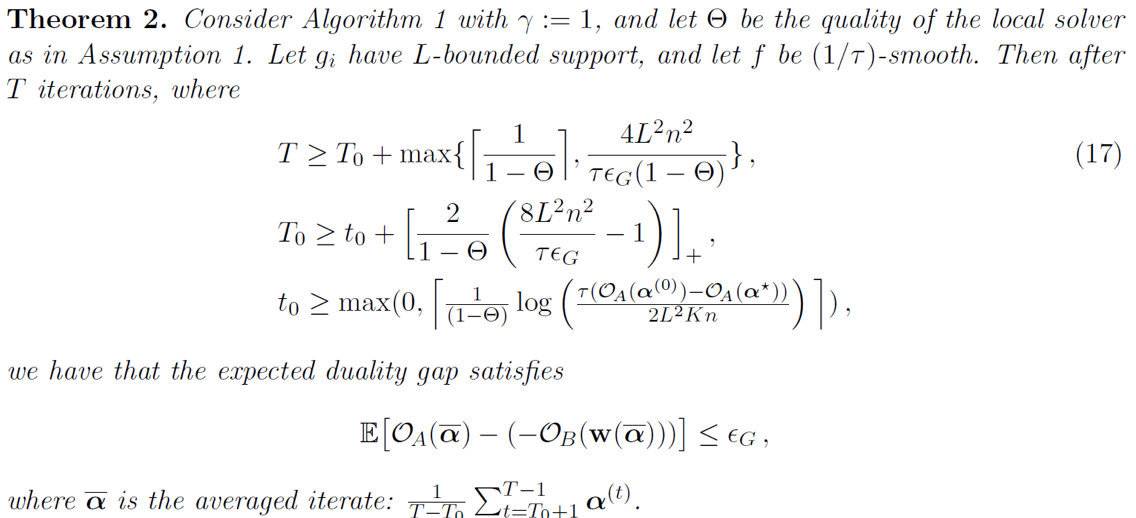
**收敛分析**

关于收敛的详细论证过于技术，会把我们的讨论弄得比较混乱，为了避免这种情况，我们这里仅给出了关键结果。感兴趣的读者可以查看原始论文详细了解。

为了简化我们的演示，这里给出了我们的三个主要假设：

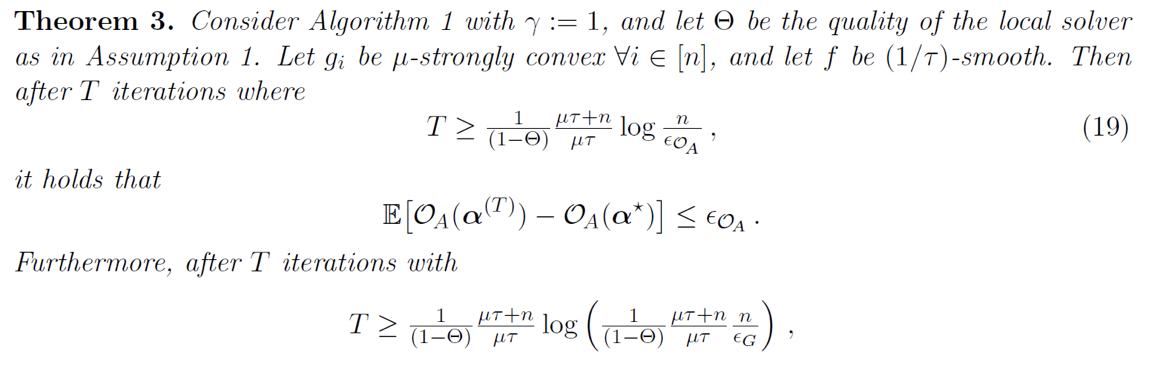
1. 数据在 K 台机器上均等分配；
2. 数据矩阵 A 的列满足 ||xi||≤1;
3. 我们仅考虑 γ=1 且 σ'=K 的情况，这能保证收敛，而且在分布式环境中的收敛速度也最快。

我们的第一个收敛结果涉及到使用一般凸性的 gi 或 L-Lipschitz 的https://image.jiqizhixin.com/uploads/wangeditor/d02dc172-e784-4a64-a74e-e35943bf6bfa/25715WX20170827-110647@2x.png（这两个条件是等价的）：



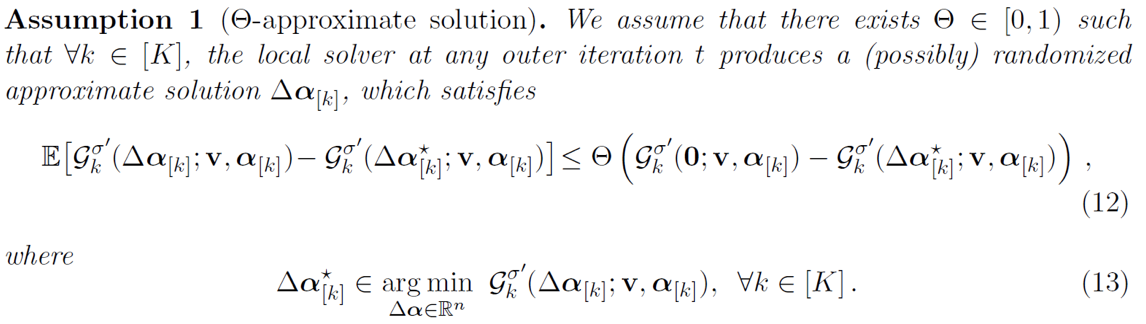
其中 L-bounded support 和https://image.jiqizhixin.com/uploads/wangeditor/d02dc172-e784-4a64-a74e-e35943bf6bfa/94131WX20170827-110800@2x.png -smooth 的定义可以在原论文找到。这个定理涵盖了带有非强凸性正则化项（比如 lasso 和稀疏 logistic 回归）的模型或带有非平滑损失（比如 SVM 的 hinge loss）的模型。

我们还可以证明在强凸性的 gi 或平滑的https://image.jiqizhixin.com/uploads/wangeditor/d02dc172-e784-4a64-a74e-e35943bf6bfa/19197WX20170827-110830@2x.png（这两个条件也是等价的）上有更快的线性收敛速度，这涵盖了弹性网络回归和 logistic 回归：



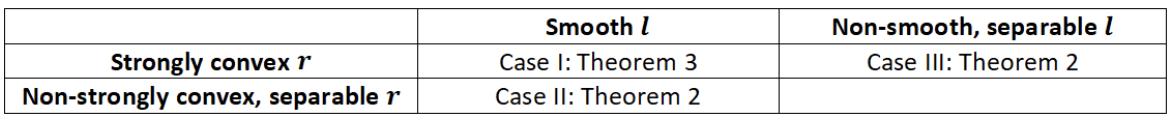
类似地，μ-strong convexity 的定义也可以在原论文找到。

这两个定理都参考了下列假设作为局部解 Θ 的质量的定义。



这个假设基本上将 Θ 定义成了局部二次问题的经验绝对偏差之前的相乘的常数。实际情况下，并行地分配到局部计算的时间应该差不多等于池化所有 K 个工作机器的更新的全部通信时间成本。

将这些收敛定理与我们前面的类别关联起来：



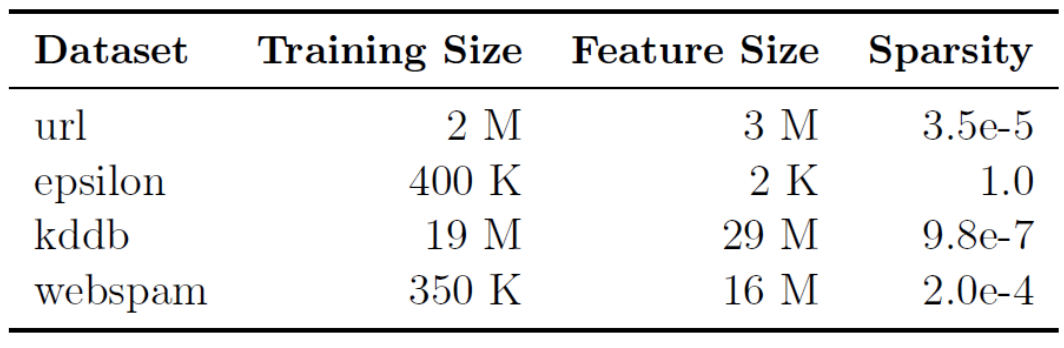
**实验**

我们将 CoCoA 与几种适用于 lasso、弹性网络回归和 SVM 的当前最佳的通用大规模分布式优化算法进行了比较：

1. MB-SGD：minibatch 随机梯度下降。对于 lasso，我们在 L1-prox 上与 MB-SGD 进行了比较。我们在 Apache Spark MLlib v1.5.0 中进行了实现和优化。
2. GD：完全梯度下降。对于 lasso，我们使用了近似版本 PROX-GD。我们在 Apache Spark MLlib v1.5.0 中进行了实现和优化。
3. L-BFGS：有限内存拟牛顿法。对于 lasso，我们使用了 OWL-QN（orthant-wise limited quasi-Newton method）。我们在 Apache Spark MLlib v1.5.0 中进行了实现和优化。
4. ADMM：交替方向乘子法。对于 lasso，我们使用了共轭梯度；对于 SVM，我们使用了 SDCA（随机对偶坐标上升）。
5. MB-CD：minibatch 并行化的坐标下降。对于 SVM，我们使用了 MB-SDCA。

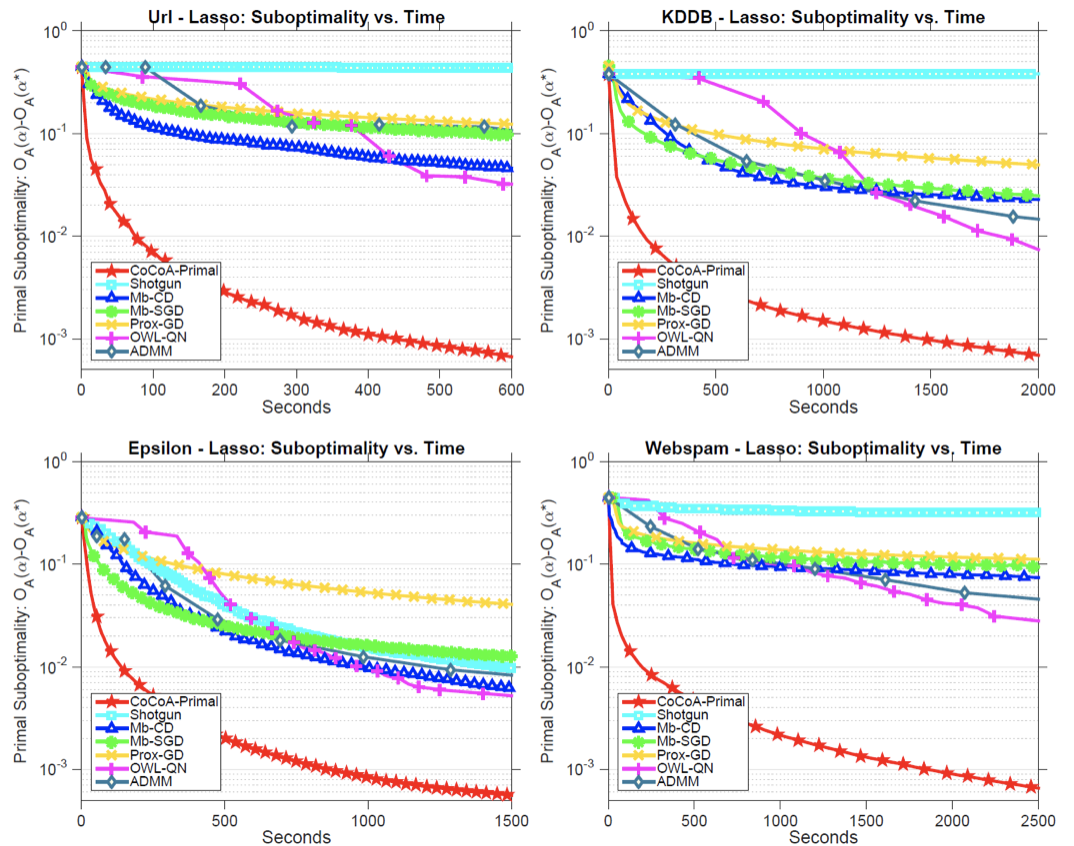
为了避免麻烦，这些不谈每种参与比较的方法的调参细节。感兴趣的读者可以参考原论文，以便重现论文的结果。对于 CoCoA，所有的实验在单机上都使用了随机坐标下降作为局部求解器。如果使用更加复杂的求解器，当然有可能进一步提升表现水平。这个开放性的实践就留给感兴趣的读者探索吧。

我们比较的指标是离原始最优（primal optimality）的距离。为此我们为所有方法都运行了大量迭代次数，直到观察不到明显的进展为止，然后选择其中最小的原始值。使用的数据集是：



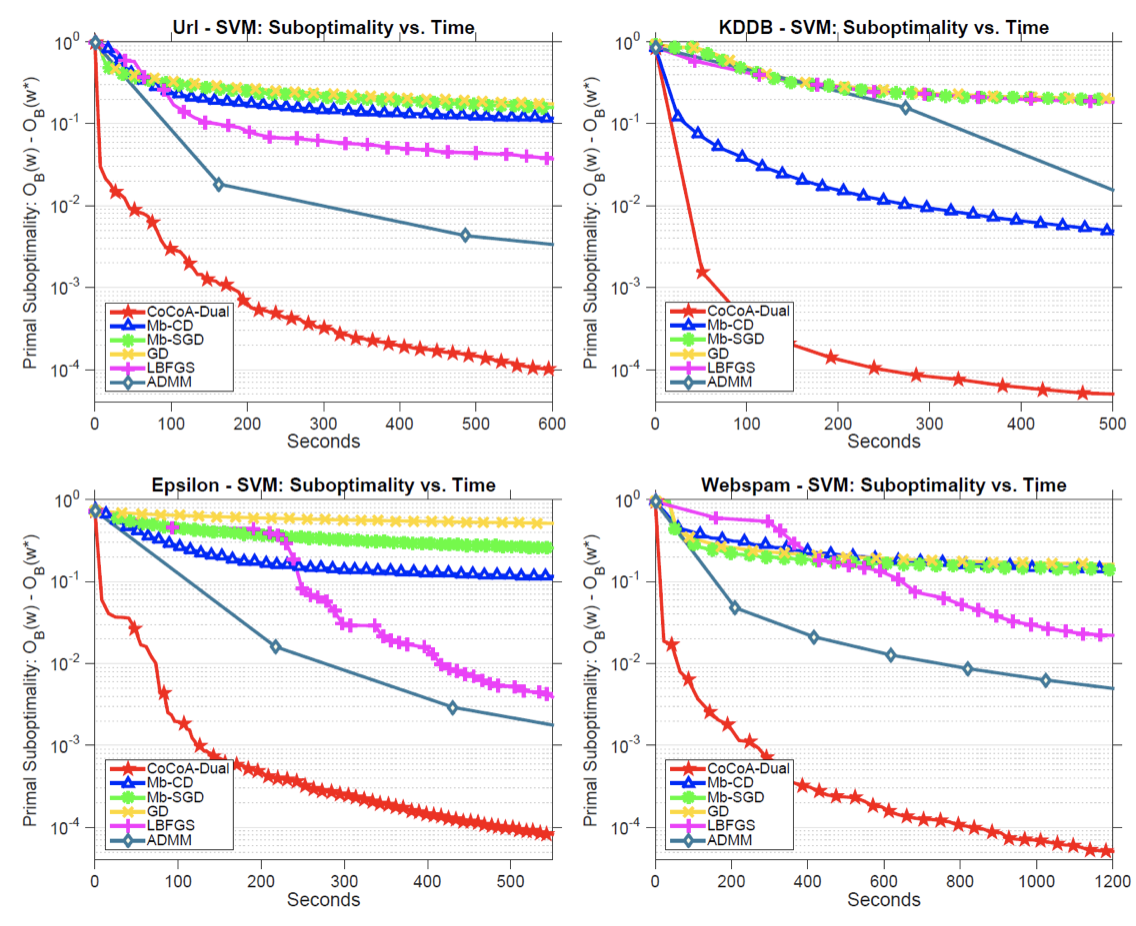
所有代码都是用 Apache Spark 编写的，并且都运行在 Amazon EC2 m3.xlarge 实例上（每台机器一个核）。代码已发布到 GitHub：[www.github.com/gingsmith/proxcocoa](https://www.jiqizhixin.com/articles/www.github.com/gingsmith/proxcocoa" \t "_blank)

在原始的设置中，我们应用 CoCoA 在上述每个数据集上拟合 lasso 模型，其中使用了随机坐标下降作为局部求解器，而总迭代次数 H 调节了局部解的质量 Θ。我们也包括了多核 SHOTGUN 作为极端案例。对于 MB-CD、SHOTGUN 和原始 CoCoA，数据集是按特征分布的；而对于 MB-SGD、PROX-GD、OWL-QN 和 ADMM，数据集则是按数据点分布的。以秒为单位绘制原始次优性，我们得到：

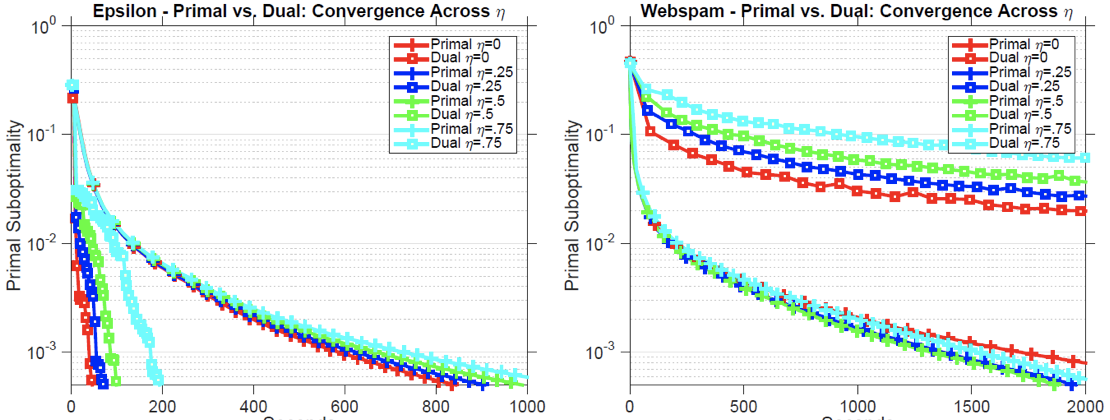


很显然，就算与比较中最好的方法 OWL-QN 相比，CoCoA 的收敛速度也快了 50 多倍，而且在有大量特征的数据集上表现最好，而这也正是 lasso 常常应用的领域。

在对偶的设置中，我们考虑的是拟合 SVM。CoCoA 使用随机对偶坐标上升作为局部求解器。所有方法都按数据点分布数据。显然，CoCoA 的表现又超出了其它方法一大截。



为了理解 CoCoA 的原始-对偶互换性，我们让其两种变体都拟合了弹性网络回归模型，并使用了坐标下降作为局部求解器。



当数据集有大量特征而非数据点时，原始 CoCoA 的表现更好，而且对强凸性的退化（deterioration）是稳健的。另一方面，当数据集有大量数据点而非特征时，对偶 CoCoA 的表现更好，在针对强凸性的损失的稳健性方面不够好。这提醒实践者在面临不同的问题设置时，应该使用不同的 CoCoA 变体——算法 2 或算法 3.

原论文还报告了更多有趣的发现，比如原始 CoCoA 可以保留局部稀疏性，并会将其最终传递为全局稀疏性；调节控制 Θ 的 H 允许机器学习系统设计者一路探索「计算-通信」权衡曲线，以确定他们当前系统的最佳平衡所在。

**总结**

CoCoA 是一个通用分布式优化框架，可以在分布式集群中实现通信高效的原始-对偶优化。它的方式是利用对偶性将全局目标分解成局部二次近似子问题，而这些子问题可以使用架构师选择的任意当前最佳的单机求解器并行地求解到任意准确度。CoCoA 的灵活性允许机器学习系统设计者和算法开发者轻松探索分布式系统的「计算-通信」权衡曲线，并为他们的特定硬件配置和计算负载选择最优的平衡。在实验中，CoCoA 将这种选择总结归纳成了单个可调的超参数 H（迭代的总次数），它间接等效的 Θ（局部解的质量）进入了关于原始和对偶 CoCoA 的收敛速度的两个重要理论证明。实证结果表明 CoCoA 的表现可以以最高 50 倍的差距超越当前最佳的分布式优化方法。

**参考文献**

[1] CoCoA: A General Framework for Communication-Efficient Distributed Optimization, V. Smith, S. Forte, C. Ma, M. Takac, M. I. Jordan, M. Jaggi, https://arxiv.org/abs/1611.02189

[2] Distributed Optimization with Arbitrary Local Solvers, C. Ma, J. Konecny, M. Jaggi, V. Smith, M. I. Jordan, P. Richtarik, M. Takac, Optimization Methods and Software, 2017, http://www.tandfonline.com/doi/full/10.1080/10556788.2016.1278445

[3] Adding vs. Averaging in Distributed Primal-Dual Optimization, C. Ma\*, V. Smith\*, M. Jaggi, M. I. Jordan, P. Richtarik, M. Takac, International Conference on Machine Learning (ICML '15), http://proceedings.mlr.press/v37/mab15.pdf

[4] Communication-Efficient Distributed Dual Coordinate Ascent, M. Jaggi\*, V. Smith\*, M. Takac, J. Terhorst, S. Krishnan, T. Hofmann, M. I. Jordan, Neural Information Processing Systems (NIPS '14), https://people.eecs.berkeley.edu/~vsmith/docs/cocoa.pdf

[5] L1-Regularized Distributed Optimization: A Communication-Efficient Primal-Dual Framework, V. Smith, S. Forte, M. I. Jordan, M. Jaggi, ML Systems Workshop at International Conference on Machine Learning (ICML '16), http://arxiv.org/abs/1512.04011