Appendix A
Correlation of temperature

We find that correlation of temperature among data centers that are usually far apart from each other is rather mild. Figure 1 shows a scatter plot of pairwise temperature correlation coefficients for all 13 locations. A few pairs are negatively correlated, and most lie in between the 0.6 and -0.6 correlation lines.

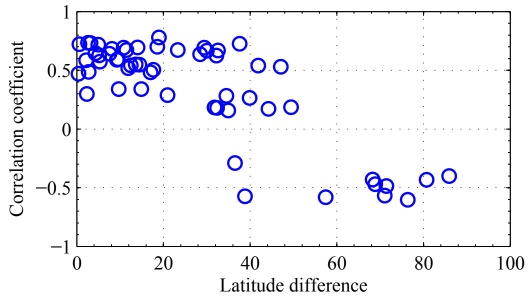


Fig. 1: The relationship between correlation coefficients of hourly temperatures and latitude for 13 Google data center locations.

Appendix B
The pPUE function

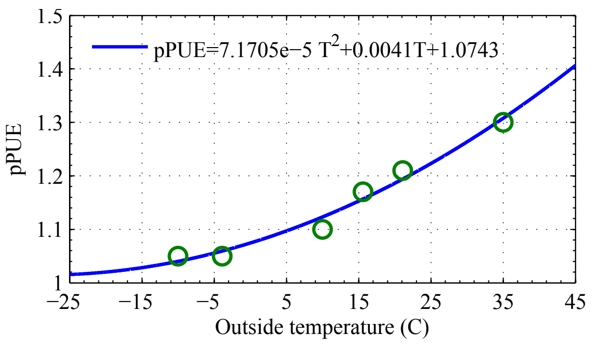


Fig. 2: Model fitting of pPUE as a function of the outside temperature T for Emerson's DSE™ CRAC [5]. Small circles denote empirical data points.

The curve can be calibrated given access to more data from measurements. For the purpose of this paper, our approach yields a tractable model that captures the overall CRAC efficiency for the entire spectrum of its operating modes. This is the one of the earliest empirical models for CRACs with both outside air and mechanical cooling, which has been increasingly adopted in the industry. Our model is also useful for future studies on data center cooling energy.

Appendix C
Discussion of the utility model

Our utility loss model is simplified and resembles (at best) part of the ground truth. Empirical verification of the model against measurement data will certainly increase the value of this work. That being said, we would like to clarify that in this paper, our main purpose is to provide a general methodology (for companies like Google to perform workload management) rather than developing particular mathematical models for performance analysis. Therefore, we use the fundamental concept of utility as in economics and game theory, and follow the basic approach of making this utility loss function as general as possible by only imposing some "natural" requirements, such as concavity. To provide concrete examples, and also to conduct performance evaluation, we provide two exemplary definitions of this function in the paper.

For the utility loss model of interactive requests, we consider three factors, D_i the demand of user i, $\{L_{ij}\}$ the latency between user i and different data centers j, and $\{\alpha_{ij}\}$ the request routing decision for user i. The reason is that for interactive services, latency is arguably the most important performance metric for users. Amazon found every 100ms of latency costs them 1% in revenues. Google found an extra 0.5s in search page generation time dropped traffic by 20%. This anecdotal evidence provides strong support for our assumption. A user's latency is clearly affected by the three parameters that we consider here. Thus our model captures the most important factors related to user's utility of accessing the service.

Appendix D
Data locality

Adding data locality constraints to the problem does not affect the complexity of solving it. At a glance, for each user, a constraint needs to be added to specify the set of data centers that have this user's data, and the number of constraints is equal to the number of users. However the problem remains convex, and the same decomposition technique can be applied to obtain per-user problems, with an additional locality constraint. Now, the only difference is that there is a locality constraint that limits the set of data centers requests can be directed to. Clearly this will not affect the complexity at all.

Appendix E
Proof of inequality (18)

Taking ∇_{x_i} on both sides of (15) in the main text, we get

$$\nabla_{x_i} L_\rho(x^k; y^k) = \nabla f_i(x_i^k) + A_i^T y^k + \rho A_i^T \left(\sum_{i=1}^{m} A_i x_i^k - b \right).$$

Recall that x_i^{k+1} minimizes $L_\rho(x_1^{k+1}, \ldots, x_{i-1}^{k+1}, x_i, x_{i+1}^k, \ldots, x_m^k; y^k)$, so we have

$$0 = \nabla f_i(x_i^{k+1}) + A_i^T y^k + \rho A_i^T \left(\sum_{j=1}^{i} A_j x_j^{k+1} + \sum_{j=i+1}^{m} A_j x_j^k - b \right).$$

Combining the two equalities above, we obtain

$$\nabla_{x_i} L_\rho(x^k; y^k) = \nabla f_i(x_i^k) - \nabla f_i(x_i^{k+1}) + \rho A_i^T \left(\sum_{j=1}^{i} A_j (x_j^k - x_j^{k+1}) \right).$$

This implies that

$$\begin{aligned} \|\nabla_{x_i} L_\rho(x^k; y^k)\|_2 &\le \|\nabla f_i(x_i^{k+1}) - \nabla f_i(x_i^k)\|_2 \\ &\quad + \sum_{j=1}^{i} \|\rho A_i^T A_j (x_j^k - x_j^{k+1})\|_2 \\ &\le \|\nabla f_i(x_i^{k+1}) - \nabla f_i(x_i^k)\|_2 \\ &\quad + \sum_{j=1}^{i} \|\rho A_i^T A_j\|_2 \|x_j^k - x_j^{k+1}\|_2, \qquad (1) \end{aligned}$$

where the last inequality follows from the definition of the matrix norm.

Recall that, by Assumption 2 in the main text, we have

$$\|\nabla f_i(x_i^{k+1}) - \nabla f_i(x_i^k)\|_2 \le \kappa_i \|x_i^k - x_i^{k+1}\|_2.$$

Substituting this in (1) gives

$$\|\nabla_{x_i} L_\rho(x^k; y^k)\|_2 \le \kappa_i \|x_i^k - x_i^{k+1}\|_2 + \sum_{j=1}^{i} \|\rho A_i^T A_j\|_2 \|x_j^k - x_j^{k+1}\|_2.$$

Since $\|x_j^k - x_j^{k+1}\|_2 \le \|x^k - x^{k+1}\|_2$ for all j, we have

$$\|\nabla_{x_i} L_\rho(x^k; y^k)\|_2 \le \theta \|x^k - x^{k+1}\|_2$$

for some $\theta > 0$. In particular, if we choose

$$\theta = \max_i \left\{ \kappa_i + \sum_{j=1}^{i} \|\rho A_i^T A_j\|_2 \right\},$$

then $\|\nabla_{x_i} L_\rho(x^k;y^k)\|_2 \le \theta\|x^k - x^{k+1}\|_2$ for all i, which implies that

$$\|\nabla_x L_\rho(x^k;y^k)\|_2^2 \le \sum_{i=1}^m \|\nabla_{x_i} L_\rho(x^k;y^k)\|_2^2 \le m\theta^2\|x^k - x^{k+1}\|_2^2.$$

Appendix F Proof of inequality (17)

This inequality is proved in Lemma 2.2 under three assumptions in pp.5 of [7]. Since these assumptions are valid in our case as well, we omit the detailed proof here.

Appendix G Proof of inequality (16)

We first introduce two lemmas that bound the changes in Δ_d^k and Δ_p^k over one iteration.

Lemma 1:

$$\Delta_d^k - \Delta_d^{k-1} \le -\varrho(Ax^k - b)^T(A\bar{x}^{k+1} - b). \tag{2}$$

Proof: By definition, $\bar{x}^{k+1}$ minimizes $L_\rho(x;y^k)$, *i.e.*,

$$L_\rho(\bar{x}^{k+1};y^k) = d(y^k).$$

Thus, we have

$$\begin{aligned}\Delta_d^k - \Delta_d^{k-1} &= \Big(d^* - d(y^k)\Big) - \Big(d^* - d(y^{k-1})\Big)\\ &= d(y^{k-1}) - d(y^k)\\ &= L_\rho(\bar{x}^k;y^{k-1}) - L_\rho(\bar{x}^{k+1};y^k)\\ &= \Big(L_\rho(\bar{x}^{k+1};y^{k-1}) - L_\rho(\bar{x}^{k+1};y^k)\Big)\\ &\quad + \Big(L_\rho(\bar{x}^k;y^{k-1}) - L_\rho(\bar{x}^{k+1};y^{k-1})\Big)\\ &= (y^{k-1} - y^k)^T(A\bar{x}^{k+1} - b)\\ &\quad + \Big(L_\rho(\bar{x}^k;y^{k-1}) - L_\rho(\bar{x}^{k+1};y^{k-1})\Big)\\ &= -\varrho(Ax^k - b)^T(A\bar{x}^{k+1} - b)\\ &\quad + \Big(L_\rho(\bar{x}^k;y^{k-1}) - L_\rho(\bar{x}^{k+1};y^{k-1})\Big)\\ &\le -\varrho(Ax^k - b)^T(A\bar{x}^{k+1} - b),\end{aligned}$$

where the last inequality follows from the fact that $\bar{x}^k$ minimizes $L_\rho(x;y^{k-1})$. □

Lemma 2:

$$\Delta_p^k - \Delta_p^{k-1} \le \varrho\|Ax^k - b\|_2^2 - \gamma\|x^{k+1} - x^k\|_2^2 - \varrho(Ax^k - b)^T(A\bar{x}^{k+1} - b). \tag{3}$$

Proof: First, we have

$$\begin{aligned}&\Delta_p^k - \Delta_p^{k-1}\\ &= \Big(L_\rho(x^{k+1};y^k) - d(y^k)\Big) - \Big(L_\rho(x^k;y^{k-1}) - d(y^{k-1})\Big)\\ &= \Big(L_\rho(x^{k+1};y^k) - L_\rho(x^k;y^{k-1})\Big) + \Big(d(y^{k-1}) - d(y^k)\Big)\\ &\le \Big(L_\rho(x^{k+1};y^k) - L_\rho(x^k;y^{k-1})\Big) - \varrho(Ax^k - b)^T(A\bar{x}^{k+1} - b),\end{aligned} \tag{4}$$

where the last inequality follows from (2).

We next bound the term $L_\rho(x^{k+1};y^k) - L_\rho(x^k;y^{k-1})$. Note that

$$\begin{aligned}L_\rho(x^k;y^k) - L_\rho(x^k;y^{k-1}) &= (y^k - y^{k-1})^T(Ax^k - b)\\ &= (\varrho(Ax^k - b))^T(Ax^k - b)\\ &= \varrho\|Ax^k - b\|_2^2.\end{aligned} \tag{5}$$

Note also that, by Assumption 2, the augmented Lagrangian

$$L_\rho(x_1,\ldots,x_m;y) = \sum_{i=1}^m f_i(x_i) + y^T(\sum_{i=1}^m A_i x_i - b) + (\rho/2)\|\sum_{i=1}^m A_i x_i - b\|_2^2$$

is strongly convex over each variable x_i, as the sum of a strongly convex function and a convex function is strongly convex. Thus, we have

$$\begin{aligned}&L_\rho(x_1^{k+1},\ldots,x_{i-1}^{k+1},x_i^k,x_{i+1}^k,\ldots,x_m^k;y^k)\\ &\quad - L_\rho(x_1^{k+1},\ldots,x_{i-1}^{k+1},x_i^{k+1},x_{i+1}^k,\ldots,x_m^k;y^k)\\ &\ge \nabla_{x_i} L_\rho(x_1^{k+1},\ldots,x_{i-1}^{k+1},x_i^{k+1},x_{i+1}^k,\ldots,x_m^k;y^k)\\ &\quad + \frac{\nu_i}{2}\|x_i^k - x_i^{k+1}\|_2^2\\ &= \frac{\nu_i}{2}\|x_i^k - x_i^{k+1}\|_2^2,\end{aligned}$$

for $i = 1,\ldots,m$, where the last equality follows from the fact that x_i^{k+1} minimizes $L_\rho(x_1^{k+1},\ldots,x_{i-1}^{k+1},x_i,x_{i+1}^k,\ldots,x_m^k;y^k)$.

Adding all the inequalities above together, we obtain

$$L_\rho(x^k;y^k) - L_\rho(x^{k+1};y^k) \ge \sum_{i=1}^m \frac{\nu_i}{2}\|x_i^k - x_i^{k+1}\|_2^2.$$

If we choose $\gamma = \min_i\{\nu_i/2\}$, then we have

$$L_\rho(x^k;y^k) - L_\rho(x^{k+1};y^k) \ge \gamma\|x^k - x^{k+1}\|_2^2,$$

or equivalently,

$$L_\rho(x^{k+1};y^k) - L_\rho(x^k;y^k) \le -\gamma\|x^{k+1} - x^k\|_2^2. \tag{6}$$

Adding (5) and (6), we obtain

$$L_\rho(x^k;y^k) - L_\rho(x^k;y^{k-1}) \le \varrho\|Ax^k - b\|_2^2 - \gamma\|x^{k+1} - x^k\|_2^2.$$

Substituting this in the first term of (4), we get

$$\Delta_p^k - \Delta_p^{k-1} \le \varrho\|Ax^k - b\|_2^2 - \gamma\|x^{k+1} - x^k\|_2^2 - \varrho(Ax^k - b)^T(A\bar{x}^{k+1} - b).$$

□

Now we are ready to prove the inequality (16) in the main text. Adding (2) and (3) gives

$$V^k - V^{k-1} \le \varrho\|Ax^k - b\|_2^2 - \gamma\|x^{k+1} - x^k\|_2^2 - 2\varrho(Ax^k - b)^T(A\bar{x}^{k+1} - b). \tag{7}$$

Since $Ax^k - A\bar{x}^{k+1} = (Ax^k - b) - (A\bar{x}^{k+1} - b)$, we have

$$\begin{aligned}\|Ax^k - A\bar{x}^{k+1}\|_2^2 &= \|(Ax^k - b) - (A\bar{x}^{k+1} - b)\|_2^2\\ &= \|Ax^k - b\|_2^2 - 2(Ax^k - b)^T(A\bar{x}^{k+1} - b)\\ &\quad + \|A\bar{x}^{k+1} - b\|_2^2.\end{aligned}$$

Substituting this in (7) yields

$$V^k - V^{k-1} \le \varrho\|Ax^k - A\bar{x}^{k+1}\|_2^2 - \varrho\|A\bar{x}^{k+1} - b\|_2^2 - \gamma\|x^{k+1} - x^k\|_2^2. \tag{8}$$

Note that

$$\begin{aligned}\|Ax^k - A\bar{x}^{k+1}\|_2^2 &\le \|A\|_2^2\|x^k - \bar{x}^{k+1}\|_2^2 &(9)\\ &\le \tau^2\|A\|_2^2\|\nabla_x L_\rho(x^k;y^k)\|_2^2 &(10)\\ &\le \tau^2\eta^2\|A\|_2^2\|x^k - x^{k+1}\|_2^2, &(11)\end{aligned}$$

where (9) follows from the definition of the matrix norm, (10) follows from (17) in the main text, and (11) follows from (18) in the main text.

Substituting this in the first term of (8), we obtain

$$\begin{aligned}V^k - V^{k-1} &\le (\varrho\tau^2\eta^2\|A\|_2^2 - \gamma)\|x^{k+1} - x^k\|_2^2 - \varrho\|A\bar{x}^{k+1} - b\|_2^2\\ &= -\varrho\|A\bar{x}^{k+1} - b\|_2^2 - \vartheta\|x^{k+1} - x^k\|_2^2,\end{aligned}$$

where $\vartheta = \gamma - \varrho\tau^2\eta^2\|A\|_2^2$. Note that, when the stepsize ϱ is small enough, we have $\vartheta > 0$, which gives (16) in the main text.

Appendix H
Per-datacenter sub-problem is a SOCP

Note that (23) in the main text can be rewritten as the following quadratic program:

$$\begin{aligned} \min \quad & a_j^T F_j a_j + h_j^T a_j \\ \text{s.t.} \quad & a_j \succeq 0, \end{aligned}$$

where $F_j = (\rho/2)\left(I_{|\mathcal{I}|} + e^T e\right)$ with $e = (1, \ldots, 1)$, and h_j is a column vector with $h_{ji} = E_j P_j + \lambda_j^k + \varphi_{ij}^k + \rho(\beta_j^k + \gamma_j^k - C_j - \alpha_{ij}^{k+1})$. Clearly, F_j captures the quadratic terms and h_j captures the linear terms in the objective function.

Since F_j is symmetric and positive-definite, F_j can be decomposed as $F_j = G_j^T G_j$ (known as the Cholesky decomposition), where G_j is an upper triangular matrix with positive diagonal entries. In particular, G_j is invertible. Let $g_j = (1/2)(G_j^{-1})^T h_j$. Then the objective function can be expressed as

$$\|G_j a_j + g_j\|_2^2 - g_j^T g_j.$$

Thus, the quadratic program is equivalent to the following second-order cone program (SOCP)

$$\begin{aligned} \min \quad & \|G_j a_j + g_j\|_2 \\ \text{s.t.} \quad & a_j \succeq 0. \end{aligned}$$

Since SOCP can be efficiently solved (using, for example, interior point methods), our per-datacenter sub-problem admits fast algorithms.

Appendix I
More discussions on our distributed 4-block ADMM in Sec. 5

For more speed-up, our algorithm can be terminated before convergence is achieved. This is a feature of ADMM as it is not sensitive to step sizes, and usually finds a solution with modest accuracy within tens of iterations [4]. An early-braking mechanism may be safely applied to terminate the algorithm after a certain number of iterations without unpredictable performance loss. This further highlights the practicality of the algorithm. The message passing overhead is low for data center interconnects that are designed to handle bulky data transfers [2].

Convergence of our algorithm. It only remains to show that the convergence result in Sec. 4 holds for our problem. Based on Theorem 1, it suffices to examine if the three assumptions hold for our problem. Assumption 2 clearly holds for convex and differentiable utility loss functions. Assumption 1 requires the objective function to be strongly convex, which may or may not hold depending on the specific form of the utility loss functions. When the utility loss of capacity allocation $V_j(\beta_j)$ is modeled by a log function as in (3) in the main text, $V(\beta) = \sum_j V_j(\beta_j)$ is strongly convex, because its Hessian matrix

$$\nabla^2 V(\beta) = r \begin{pmatrix} 1/\beta_1^2 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & 1/\beta_{|\mathcal{J}|}^2 \end{pmatrix}$$

satisfies $\nabla^2 V(\beta) - \nu I_{|\mathcal{I}|} \succeq 0$, where $\nu = \min_j \{r/C_j^2\}$. When the utility loss of request routing $U_i(\alpha_i)$ is modeled by a quadratic function of the average latency as in (2) of the main text, $U_i(\alpha_i)$ is not strongly convex. Nevertheless, $U_i(\alpha_i)$ can be well approximated by

$$\bar{U}_i(\alpha_i) = U_i(\alpha_i) - \epsilon \sum_j \left(\frac{\alpha_{ij}}{D_i}\right) \log\left(\frac{D_i}{\alpha_{ij}}\right)$$

for some sufficiently small $\epsilon > 0$. It is easily verified that $\bar{U}_i(\alpha_i)$ is strongly convex with constant ϵ/D_i. Moreover, there is an engineering advantage of such an approximation. Note that the entropy of a request routing scheme α_i is given by $\sum_j \left(\frac{\alpha_{ij}}{D_i}\right) \log\left(\frac{D_i}{\alpha_{ij}}\right)$. The larger the entropy is, the better the request routing in terms of the diversity. Thus, the second term in $\bar{U}_i(\alpha_i)$ tends to increase the entropy, which in turn improves the diversity.

Appendix J
More on simulation setup

We use the 2011 annual average day-ahead on peak prices [6] at the local markets as the power prices P_j for the 6 U.S. locations[1]. For non-U.S. locations, the power price is calculated based on the retail industrial power price available on the local utility company websites with a 50% wholesale discount. Table 1 lists the power prices at each location.

Council Bluffs, IA	42.73	Berkeley County, SC	44.44
The Dalles, OR	32.57	Lenoir, NC	40.68
Mayes County, OK	36.41	Douglas County, GA	39.97
Quilicura, Chile	75.69	St. Ghislain, Belgium	50.50
Hamina, Finland	43.84	Dublin, Ireland	50.62
Hong Kong	36.12	Taiwan	31
Singapore	66.72		

TABLE 1: Power prices ($USD/MWh) at different locations.

To calculate the utility loss of interactive workloads, we rely on iPlane [8], a system that collects wide-area network statistics from Planetlab vantage points, to obtain the latency matrix L. Since the Wikipedia traces do not contain client side information, we emulate the geographical diversity of user requests by splitting the total interactive workloads among users following a normal distribution. We set the number of users $|\mathcal{I}| = 10^5$, and choose 10^5 IP prefixes from a RouteViews [1] dump. We then extract the corresponding round trip times from iPlane logs, which contain traceroutes made to IP addresses from Planetlab nodes. We only use latency measurements from Planetlab nodes that are close to our data center locations to resemble the user-data center latency.

Appendix K
Latency performance

Figure 3 shows the average latency of all approaches with the settings described in Sec. 6.2.

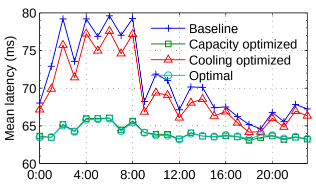


Fig. 3: *Baseline* and *Cooling optimized* induce larger average latency.

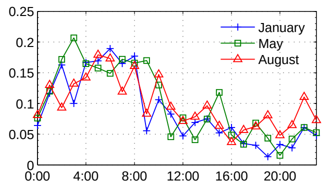
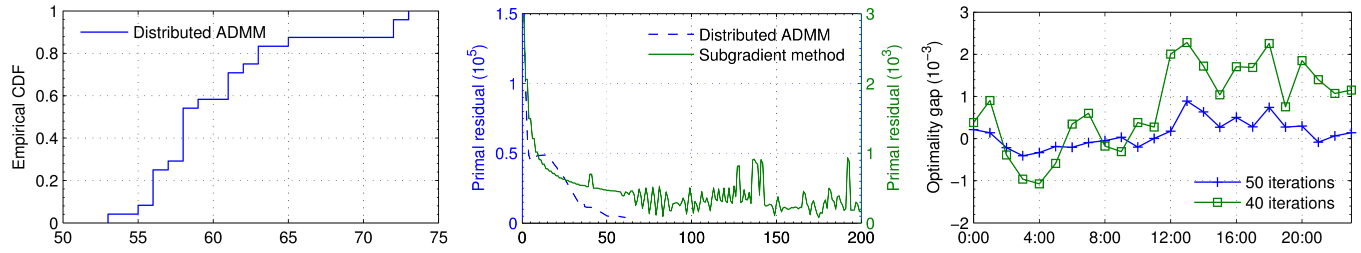


Fig. 4: Overall cost saving is insensitive to seasonal changes of the climate.

Appendix L
Sensitivity to seasonal changes

One natural question is, since the results above are obtained in winter times (January), would the benefits be less significant during summer times when cooling is more expensive? In other words, are the benefits sensitive to the seasonal changes? We thus run our *Optimal* approach with *Baseline* at each day of May, which represents typical Spring/Fall weather, and August, which represents typical Summer weather, respectively. Figure 4 shows the average overall cost savings achieved in different seasons. We observe that the cost savings, ranging from 5% to 20%, are consistent and insensitive to seasonal changes. The reason is that our approach depends on: 1) the geographical diversity of temperature and cooling efficiency; 2) the mixed nature of data center workloads, both of which exist at all times of the year no matter which cooling method is used. Temperature aware workload management is thus expected to offer consistent cost benefits.

1. The U.S. electricity market is consisted of multiple regional markets. Each regional market has several hubs with their own pricing. We thus use the price of the specific hub that each U.S. data center locates in.



(a) CDF of convergence iterations. (b) Primal residuals of algorithms. (c) Early braking at 50 and 40 iterations.

Fig. 5: Convergence results of our distributed ADMM algorithm compared against subgradient methods.

Appendix M Algorithm Convergence

We evaluate the convergence of our distributed ADMM algorithm in this section. As a benchmark, a dual decomposition approach is used to tackle the original optimization (6) in the main text, with the standard Lagrangian

$$L(\alpha,\beta;\lambda)=\sum_j E(\sum_i \alpha_{ij})P_j+\sum_i U_i(\alpha_i)$$
$$+\sum_j (E(\beta_j)P_j+V_j(\beta_j))+\sum_j \lambda_j(\sum_i \alpha_{ij}+\beta_j-C_j).$$

It can be readily seen that minimizing $L(\alpha,\beta;\lambda)$ can be separately done over α and β since the energy cost function E is linear. The α- and β-minimization problems can also be decomposed into per-user and per-data center sub-problems. The dual variable λ is updated with subgradient methods [3] as follows:

$$\lambda_j^{k+1}=\max\Big\{0,\lambda_j^k+\varrho^k\Big(\sum_i \alpha_{ij}+\beta_j-C_j\Big)\Big\},$$

where ϱ is the step size. We optimize the step size $\varrho^k=10^{-6}/\sqrt{k}$ according to the diminishing step size rule [3]. For our distributed ADMM algorithm, the penalty parameter $\rho=3\times10^{-7}$, and the step size $\varrho=10^{-6}$.

The stopping rules of the algorithms are set as follows. ADMM algorithms are usually stopped when the primal and dual residuals are smaller than certain tolerance thresholds [4]. The calculations of primal and dual residuals and the tolerance thresholds are identical to those in [4], and we omit details here. For dual decomposition with subgradient methods, it is terminated when $\|\alpha^{k+1}-\alpha^k\|_2^2<10^{-2}\|\alpha^k\|_2^2$, or when the number of iterations exceeds 200. Other parameter setup, including the scale of the problems, is the same as in previous simulations.

Figure 5a plots the empirical cumulative distribution function (CDF) of convergence iterations for the 24 runs using our traces. We find that the dual decomposition approach with subgradient methods cannot converge within 200 iterations in all runs. Thus we only show the CDF for our algorithm. Observe that distributed ADMM converges within 73 iterations in all runs, and the fastest run uses 53 iterations only. The convergence of distributed ADMM thus significantly outperforms the traditional subgradient methods. Figure 5b depicts a sample path of the convergence of the primal residual for the two algorithms. We point out the scales of the primal residuals for the two algorithms are different, since the distributed ADMM solves the 4-block formulation (19) in the main text while the subgradient method solves the original formulation (6) in the main text. We can see that the curve of distributed ADMM decreases smoothly and reaches below the tolerance threshold after 61 iterations. The subgradient method suffers from oscillations after 70 iterations, and fails to decrease below the threshold.

Finally, Figure 5c shows the performance of early-braking for our distributed ADMM algorithm. We plot the solutions of the algorithm after 50 and 40 iterations, respectively. Clearly, the optimality gap of stopping after 40 iterations is larger than stopping after 50 iterations. A more interesting observation is that the optimality gap is strikingly small — only 10^{-3} relative to the optimum. There are times when the optimality gap becomes negative. This is caused by (primal or dual) infeasible solutions produced by early-braking during demand peak periods. The feasibility gap is rather small, though, and can be readily fixed.

The results demonstrate that the distributed ADMM algorithm converges quickly, and is better suited to large-scale convex optimization problems. The early-braking mechanism can further improve the convergence in practice with negligible performance loss.

References

[1] http://www.routeviews.org.

[2] M. Al-Fares, A. Loukissas, and A. Vahdat. A scalable, commodity data center network architecture. In *Proc. ACM SIGCOMM*, 2008.

[3] S. Boyd and A. Mutapcic. Subgradient methods. Lecture notes of EE364b, Stanford University, Winter Quarter 2006-2007. http://www.stanford.edu/class/ee364b/notes/subgrad_method_notes.pdf.

[4] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2010.

[5] Emerson Network Power. Liebert® DSE™ precision cooling system. http://tinyurl.com/c7e8qxz, 2012.

[6] Federal Energy Regulatory Commission. U.S. electric power markets. http://www.ferc.gov/market-oversight/mkt-electric/overview.asp, 2011.

[7] M. Hong and Z.-Q. Luo. On the linear convergence of the alternating direction method of multipliers, August 2012.

[8] H. V. Madhyastha, T. Isdal, M. Piatek, C. Dixon, T. Anderson, A. Krishnamurthy, and A. Venkataramani. iPlane: An information plane for distributed services. In *Proc. USENIX OSDI*, 2006.