PDLP: A Practical First-Order Method for Large-Scale Linear Programming

Fan Tailin

Apr 15^{th}, 2025

PDHG (Primal-Dual hybrid gradient)

Consider the following primal-dual LP:

$$\min c^\top x \quad \max q^\top y + l^\top \lambda^+ - u^\top \lambda^- \tag{1}$$

$$\text{s.t. } Ax = b \quad \text{s.t. } c - K^\top y = \lambda \tag{2}$$

$$Gx \geq h \qquad y_{1:m_1} \geq 0 \tag{3}$$

$$l \leq x \leq u \qquad \lambda \in \Lambda \tag{4}$$

where $G \in \mathbb{R}^{m_1 \times n}$, $A \in \mathbb{R}^{m_2 \times n}$, $c \in \mathbb{R}^n$, $h \in \mathbb{R}^{m_1}$, $b \in \mathbb{R}^{m_2}$, $l \in (\mathbb{R} \cup \{-\infty\})^n$, $u \in (\mathbb{R} \cup \{+\infty\})^n$, $K^\top = (G^\top, A^\top)$, $q^\top = (h^\top, b^\top)$, and

$$\Lambda = \Lambda_1 \times \cdots \times \Lambda_n, \quad \Lambda_i = \begin{cases} \{0\} & l_i = -\infty, u_i = +\infty \\ \mathbb{R}^- & l_i = -\infty, u_i \in \mathbb{R} \\ \mathbb{R}^+ & l_i \in \mathbb{R}, u_i = +\infty \\ \mathbb{R} & \text{otherwise} \end{cases}$$

PDHG

Saddle-point problem

To solve it, we can consider the equivalent saddle-point problem:

$$\min_{x \in X} \max_{y \in Y} \mathcal{L}(x, y) = c^\top x - y^\top K x + q^\top y \tag{2}$$

with $X = \{x \in \mathbb{R}^n : l \leq x \leq u\}$ and $Y = \mathbb{R}^{m_2} \times \mathbb{R}^{m_1}_+$.

Georgia Institute of Technology

PDHG

Constrained PDHG

LP problems usually have constraints, so we need constrained PDHG.
Primal update

$$x^{t+1} = \text{proj}_X\left(x^t - \tau(c - K^\top y^t)\right)$$

Dual update

$$y^{t+1} = \text{proj}_Y\left(y^t + \sigma\left(q - K(2x^{t+1} - x^t)\right)\right).$$

τ and σ are step sizes for primal and dual. proj_Y and proj_X are projections for primal and dual, $\text{proj}_X(x) = \text{argmin}_{z\in X}\|x - z\|_2$.
Dual variable λ is not explicitly generated during PDHG, instead we compute

$$\lambda = \text{proj}_\Lambda(c - K^T y)$$

The termination for PDHG relies on duality gap

PDHG

Duality gap

Duality gap is the difference between primal and dual solutions. It can be measured with the Lagrangian:

$$\max_{\hat{z} \in Z} \{\mathcal{L}(x, \hat{y}) - \mathcal{L}(\hat{x}, y)\}.$$

For minimization problem, the optimal duality gap is always greater or equal 0, the it equals to 0 iff the strong duality holds (which obviously holds for LP).
The saddle point problem is to find a pair of solutions $z^* = (x^*, y^*)$ that $\mathcal{L}(x^*, y) \leq \mathcal{L}(x^*, y^*) \leq \mathcal{L}(x, y^*)$. Thus for saddle point problem constructed by LP, the theoretical optimal solution will be reached when the duality gap is 0.

Georgia Institute of Technology

PDHG

Algorithm 1 PDHG-C

Initialization

Pick y^0 and a feasible $x^0 \in X$, set $k \leftarrow 0$;

Choose step sizes

Choose step sizes τ_k and θ_k;

repeat

  Updating

  $x^{k+1} = \text{proj}_X \left(x^k - \tau(c - K^\top y^k)\right)$;

  $y^{k+1} = \text{proj}_Y \left(y^k + \sigma \left(q - K(2x^{k+1} - x^k)\right)\right)$;

  Termination check

  $k \leftarrow k + 1$;

until a stopping criterion is satisfied;

Output: x^k, y^k.

PDLP

PDHG norm and step reparameterization

The step sizes τ and σ can be reparameterization as

$$\tau = \eta/\omega, \sigma = \eta\omega,$$

with $\eta \in (0, \infty)$ and $\omega \in (0, \infty)$. We call ω the primal weight, and η the step size now.
The PDHG norm then can be defined with primal weight ω:

$$\|z\|_\omega := \sqrt{\omega\|x\|_2^2 + \frac{\|y\|_2^2}{\omega}},$$

where z is the primal-dual vector (x, y). The PDHG norm plays a role in theory for the PDHG and practice for the PDLP.

Georgia Institute of Technology

PDLP

Algorithm 2 PDLP (after preconditioning and presolve)

Input: An initial solution $z^{0,0}$.
Initialize outer loop counter $n \leftarrow 0$, total iterations $k \leftarrow 0$, step size $\hat{\eta}^{0,0} \leftarrow 1/\|K\|_\infty$, primal weight $\omega^0 \leftarrow \text{InitializePrimalWeight}(c, q)$;
repeat
    $t \leftarrow 0$;
    repeat
        $z^{n,t+1}, \eta^{n,t+1}, \hat{\eta}^{n,t+1} \leftarrow \text{AdaptiveStepOfPDHG}(z^{n,t}, \omega^n, \hat{\eta}^{n,t}, k)$;
        $\bar{z}^{n,t+1} \leftarrow \frac{1}{\sum_{i=1}^{t+1} \eta^{n,i}} \sum_{i=1}^{t+1} \eta^{n,i} z^{n,i}$;
        $z_c^{n,t+1} \leftarrow \text{GetRestartCandidate}(z^{n,t+1}, z_m^{n,t+1}, z^{n,0})$;
        $t \leftarrow t+1$, $k \leftarrow k+1$;
    until restart or termination criteria holds;
    restart the outer loop $z^{n+1,0} \leftarrow z_c^{n,t}$, $n \leftarrow n+1$;
    $\omega^n \leftarrow \text{PrimalWeightUpdate}(z^{n,0}, z^{n-1,0}, \omega^{n-1})$;
until termination criteria holds;
Output: $z^{n,0}$.

Georgia Institute of Technology

PDLP

Improvements

PDLP has some improvements to achieve better practial performance compare with PDHG.

- Preconditioning
- Adaptive restarts
- Adaptive step size
- Primal weight update
- Infeasibility detection

Georgia Institute of Technology

PDLP-Preconditioning

Preconditioning is a popular heuritic in optimization for improving the convergence of FOM (First-order method) by shrinking the condition number of coefficient matrix K.

Condition number

Condition number measures how much the output value of a function can change for a small change in the input argument (sensitivity to input's errors). For a linear equation system $Ax = b$, its condition number is $k(A) = \|A\| \cdot \|A^{\dagger}\|$, the norm often takes l_2, the $A^{\dagger}$ is the Moore-Penrose pseudoinverse of matrix A.

Georgia Institute of Technology

PDLP-Preconditioning

Many preconditioning methods contain matrix factorizations. Thus PDLP only consider using diagonal preconditioners: Pock-Chambolle and Ruiz. Preconditioning create a new LP instance. With diagonal matrices D_1 and D_2, K is rescaled to $D_1 K D_2 = \tilde{K} = (\tilde{G}, \tilde{A})$,
$\tilde{x} = D_2^{-1} x,\ \tilde{c} = D_2 c,\ (\tilde{b}, \tilde{h}) = D_1(b, h),\ \tilde{u} = D_2^{-1} u,\ \tilde{l} = D_2^{-1} l.$

Pock-Chambolle

The preconditioners are defined by $(D_1)_{jj} = 1/\sqrt{\|K_{j,\cdot}\|_{2-\alpha}}$ for $j = 1, ..., m_1 + m_2$ and $(D_2)_{ii} = 1/\sqrt{\|K_{\cdot,i}\|_{\alpha}}$ for $i = 1, ..., n$. In PDLP we choose $\alpha = 1$.

Ruiz

The preconditioners are defined by $(D_1)_{jj} = 1/\sqrt{\|K_{j,\cdot}\|_{\infty}}$ for $j = 1, ..., m_1 + m_2$ and $(D_2)_{ii} = 1/\sqrt{\|K_{\cdot,i}\|_{\infty}}$ for $i = 1, ..., n$.

In particular, PDLP applies 10 iterations of Ruiz scaling and then 1 Pock-Chambolle.

PDLP-Preconditioning

Consider the ill-conditioned 2×4 matrix:

$$A = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & \epsilon & 0 & \epsilon \end{bmatrix}, \quad \epsilon = 10^{-6}.$$

- **Condition Number**: Using SVD, singular values are $\sigma_1 \approx 2$, $\sigma_2 \approx \epsilon$. Thus:

$$k(A) = \frac{\sigma_1}{\sigma_2} \approx \frac{2}{\epsilon} = 2 \times 10^6.$$

Georgia Institute of Technology

PDLP-Preconditioning

- **Pock-Chambolle Method ($\alpha = 1$)**: Preconditioners use ℓ_1-norm:

$$(D_1)_{jj} = 1/\sqrt{\|A_{j,\cdot}\|_1}, \quad (D_2)_{ii} = 1/\sqrt{\|A_{\cdot,i}\|_1}.$$

  - $D_1 = \text{diag}(1, 0.9999995, 1, 0.9999995) \approx \text{diag}(1, 1, 1, 1)$, $D_2 = \text{diag}(0.5, 707.1068)$.
  - Processed matrix: $D_2 A D_1 = \begin{bmatrix} 0.5 & 0.5 & 0.5 & 0.5 \\ 0 & 0.7071068 & 0 & 0.7071068 \end{bmatrix}$.
  - Condition number: $\sigma_1 \approx 1.3066$, $\sigma_2 \approx 0.5412$, $k \approx 2.414$ (vs. 2×10^6).
- **Ruiz Method**: Preconditioners use ℓ_∞-norm, iterative:

$$(D_1)_{jj} = 1/\sqrt{\|A_{j,\cdot}\|_\infty}, \quad (D_2)_{ii} = 1/\sqrt{\|A_{\cdot,i}\|_\infty}.$$

  - First iteration: $D_1 = \text{diag}(1, 1, 1, 1)$, $D_2 = \text{diag}(1, 1000)$.
  - Processed matrix: $D_2 A D_1 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 0.001 & 0 & 0.001 \end{bmatrix}$.
  - Condition number: $\sigma_1 \approx 2$, $\sigma_2 \approx 0.0014142$, $k \approx 1414.21$ (vs. 2×10^6).

Georgia Institute of Technology

PDLP-Adaptive step size

Convergence promise

$$\eta \leq \frac{\|z^{k+1} - z^k\|_\omega^2}{2(y^{k+1} - y^k)^T K(x^{k+1} - x^k)}$$

Step size choice

- **Classical choice**:

$$\eta = \frac{1}{\|K\|_2}$$

  PDLP will always converge if $\eta \leq \frac{1}{\|K\|_2}$. But this choice is "pessimistic" and requires the estamition of $\|K\|_2$.

- **Adaptive choice**: ALGO 3

Georgia Institute of Technology

PDLP-Adaptive step size

Algorithm 3 One step of PDHG using our step size heuristic

function AdaptiveStepOfPDHG$(z^{n,t}, w^n, \eta^{n,t}, k)$
  $(x, y) \leftarrow z^{n,t}, \eta \leftarrow \eta^{n,t}$;
  for $i \leftarrow 1, \ldots, \infty$ **do**
    $x' \leftarrow \text{proj}_X\left(x - \frac{\eta}{w^n}(c - K^T y)\right)$;
    $y' \leftarrow \text{proj}_Y\left(y + \eta w^n (q - K(2x' - x))\right)$;
    $\bar{\eta} \leftarrow \frac{\|(x-x', y-y')\|^2_{\omega^n}}{2(y'-y)^T K(x'-x)}$;
    $\eta' \leftarrow \min\left((1 - (k+1)^{-0.3})\bar{\eta}, (1 + (k+1)^{-0.6})\eta\right)$;
    if $\eta \leq \bar{\eta}$ **then**
      return $(x', y'), \eta, \eta'$;
    end if
    $\eta \leftarrow \eta'$;
  end for
end function

Georgia Institute of Technology

PDLP-Adaptive step size

Note

- η will be reduced again if $\eta > \bar{\eta}$. The ALGO will be repeated then.
- $\bar{\eta} \geq \frac{1}{\|K\|_2}$ always holds, which make $\eta \geq \frac{1-o(1)}{\|K\|_2}$ holds as $k \to \infty$.
- The exponents -0.3 and -0.6 are heuristics, likely found experimentally, ensuring decrease is more aggressive than increase, and adjustments diminish as k grows, leading to stabilization.

Georgia Institute of Technology

PDLP-Primal weight update

$$\text{InitializePrimalWeight}(c, q) := \begin{cases} \frac{\|c\|_2}{\|q\|_2} & \|c\|_2, \|q\|_2 > \epsilon_{zero} \\ 1 & otherwise \end{cases},$$

where ϵ_{zero} is a small nonzero tolerance.

Algorithm 4 Primal weight update

function PRIMALWEIGHTUPDATE($z^{n,0}, z^{n-1,0}, \omega^{n-1}$)
  $\Delta_x^n \leftarrow \|x^{n,0} - x^{n-1,0}\|_2$, $\Delta_y^n \leftarrow \|y^{n,0} - y^{n-1,0}\|_2$;
  if $\Delta_x^n > \epsilon_{\text{zero}}$ **and** $\Delta_y^n > \epsilon_{\text{zero}}$ **then**
    return $\exp\left(\theta \log\left(\frac{\Delta_y^n}{\Delta_x^n}\right) + (1-\theta)\log(\omega^{n-1})\right)$;
  else
    return ω^{n-1};
  end if
end function

PDLP-Primal weight update

Insight of the update function

Heuristically we want to adjust the primal weight ω^n such that distance optimality in the primal and dual is almost the same, i.e.

$$\|(x^{n,t} - x^*, 0)\|_\omega \approx \|(0, y^{n,t} - y^*)\|_\omega,$$

where y^* and x^* are optimal solution for dual and primal. By definition of PDHG norm, we have

$$\|(x^{n,t} - x^*, 0)\|_\omega = \sqrt{\omega}\|x^{n,t} - x^*\|_2,\ \|(0, y^{n,t} - y^*)\|_\omega = \frac{1}{\sqrt{\omega}}\|y^{n,t} - y^*\|_2.$$

To make the two terms equal yields $\omega = \|y^{n,t} - y^*\|_2 / \|x^{n,t} - x^*\|_2$. The function

$$\exp\left(\theta \log\left(\frac{\Delta_y^n}{\Delta_x^n}\right) + (1-\theta)\log(\omega^{n-1})\right)$$

estimate the ω term with Δ_x^n and Δ_y^n, and dampen the variation with log-scale and exponential smooth($\theta \in [0,1]$, PDLP takes $\theta = 0.5$).

PDLP-Infeasibility detection

Divergence behavior

When the LP is infeasible, PDLP iterates $z^k = (x^k, y^k)$ don't converge. Instead:

- They diverge towards infinity, but *along a specific direction*.
- This movement follows a **ray**: starting from some point z^* and extending infinitely in a direction $v = (v_x, v_y)$.
- The direction v is called the **infimal displacement vector**.

The infimal displacement vector

The key insight from Lu et al. (2021) connects the direction vector $v = (v_x, v_y)$ to infeasibility:

- If Primal LP is Infeasible $\iff$ Dual component $v_y \neq 0$.
- If Dual LP is Infeasible $\iff$ Primal component $v_x \neq 0$.
- If Both Feasible $\implies$ $v = (0, 0)$.

PDLP-Infeasibility detection

Detection in PDLP

PDLP periodically estimates the direction v from the iterates $z^{n,t}$ by three sequences converging to v starting form the most recent restart:

1. Difference of Iterates: $d^{n,t} = z^{n,t+1} - z^{n,t}$
2. Normalized Iterates: $(z^{n,t} - z^{n,0})/t$
3. Normalized Average: $2(\bar{z}^{n,t} - \bar{z}^{n,0})/(t+1)$

Note

Verifying all three sequences is often faster than relying on just one, since their convergence speed varies with the problem's geometry.

Georgia Institute of Technology

PDLP-Adaptive restart

Normalized duality gap

Recall that duality gap was defined as

$$\max_{\hat{z} \in Z} \{\mathcal{L}(x, \hat{y}) - \mathcal{L}(\hat{x}, y)\}.$$

However this function can lead to infinity if the problem is unbounded. So Lu et al. (2022) introduces normalized duality gap $\rho_r(z)$:

$$\rho_r(z) := \frac{\max_{\hat{z} \in W_r(z)} \{\mathcal{L}(x, \hat{y}), \mathcal{L}(\hat{x}, y)\}}{r},$$

where $W_r(z) := \{\hat{z} \in Z \mid \| z - \hat{z} \| \leq r\}$ is the ball centered at z with radius $r \in (0, \infty)$ intersected with the set Z, and $\| \cdot \|$ is a carefully selected semi-norm on Z. In PDLP, the authors redefined $\mu_n(z, z_{ref})$ as the normalized duality gap at z with radius $\|z - z_{ref}\|_{\omega^n}$, i.e.

$$\mu_n(z, z_{ref}) = \rho^n_{\|z - z_{ref}\|_{\omega^n}}(z).$$

PDLP-Adaptive restart

Restart Candidate

GetRestartCandidate$(z^{n,t+1}, \bar{z}^{n,t+1}, z^{n,0}) :=$

$$\begin{cases} z^{n,t+1} & \mu_n(z^{n,t+1}, z^{n,0}) < \mu_n(\bar{z}^{n,t+1}, z^{n,0}) \\ \bar{z}^{n,t+1} & \textit{otherwise} \end{cases}.$$

Obviously this is a heuristic for greedy step size choices.

Georgia Institute of Technology

PDLP-Adaptive restart

Restart criteria

1. **Sufficient decay in normalized duality gap**:

$$\mu_n(z_c^{n,t+1}, z^{n,0}) \leq \beta_{sufficient}\mu_n(z^{n,0}, z^{n-1,0})$$

.

2. **Necessary decay + no local progress in normalized duality gap**:

$$\mu_n(z_c^{n,t+1}, z^{n,0}) \leq \beta_{necessary}\mu_n(z^{n,0}, z^{n-1,0})$$

$$\mu_n(z_c^{n,t+1}, z^{n,0}) > \mu_n(z_c^{n,t}, z^{n,0})$$

.

3. **Long inner loop**: $t \geq \beta_{artifical} k$.

The parameters are $\beta_{necessary} \in (0,1)$, $\beta_{sufficient} \in (0, \beta_{necessary})$ and $\beta_{artificial} \in (0,1)$. PDLP uses $\beta_{necessary} = 0.9$, $\beta_{sufficient} = 0.1$ and $\beta_{artificial} = 0.5$.

PDLP-Adaptive restart

Let's break down the insight of the three criteria.

1. **Sufficient decay in normalized duality gap**: The duality gap for n iterate is considered sufficiently decayed when the progress of the n iterate is detected to have reached the $\beta_{sufficient}$ progress of the $n-1$ iterate. Restart at this point.
2. **Necessary decay + no local progress in normalized duality gap**: The duality gap for n iterate is considered necessarily decayed when the progress of the n iterate is detected to have reached the $\beta_{necessary}$ progress of the $n-1$ iterate. Restart will happen if the duality gap oscillates at this point.
3. **Long inner loop**: Restart when internal loop time is too long.

Georgia Institute of Technology

PDLP-Adaptive restart

Sharpness

For a given function f, we say f is α-sharp if it holds for any $z \in Z$ that

$$f(z) - f^* \geq \alpha \mathbf{dist}(z, Z^*)^p,$$

where Z^* is the set of optima of the function f, f^* is the optimal function value, and $\mathbf{dist}(z, Z) := \inf_{\hat{z} \in Z} \|z - \hat{z}\|$. $p = 1$ means weak sharpness, $p = 2$ means strong sharpness.
Sharpness is related to the speed of convergence near the optimal solution of function f.

Georgia Institute of Technology

PDLP-Adaptive restart

Theoretical guarantee

Lu et al. (2022) gives a restart scheme for the primal-dual class of algorithms, and also gives a convergence analysis of the algorithms based on sharpness.

- LP-constructed primal-dual problems are shown to be α-sharp within $W_R(0)$, where $W_R(0) = \{\hat{z} \in Z || \hat{z}| \leq R\} \cap Z$.
- For restarted PDHG, there exists a constant $R > 0$ such that $z^{n,0} \in W_R(z^{0,0})$ for all $n \in \mathbb{N}$.
- If the primal-dual problem α-sharp on the set S containing all $z^{n,0}$, the distance to primal-dual solution decays linearly.

Georgia Institute of Technology

PDLP-Termination criteria

- **Duality gap**:

$$|q^T y + l^T \lambda^+ - u^T \lambda^- - c^T x| \leq \epsilon(1 + |q^T y + l^T \lambda^+ - u^T \lambda^-| + |c^T x|)$$

- **Primal feasibility**:

$$\left\| \begin{pmatrix} Ax - b \\ (h - Gx)^+ \end{pmatrix} \right\|_2 \leq \epsilon(1 + \|q\|_2)$$

- **Dual feasibility**:

$$\|c - K^T y - \lambda\|_2 \leq \epsilon(1 + \|c\|_2)$$

where $\epsilon \in (0, \infty)$ is the termination tolerance. PDLP sets $\epsilon = 10^{-4}$ for moderately accurate solutions and $\epsilon = 10^{-8}$ as a benchmark for high-quality solutions.

Georgia Institute of Technology

PDLP

Algorithm 5 PDLP (after preconditioning and presolve)

Input: An initial solution $z^{0,0}$.
Initialize outer loop counter $n \leftarrow 0$, total iterations $k \leftarrow 0$, step size $\hat{\eta}^{0,0} \leftarrow 1/\|K\|_\infty$, primal weight $\omega^0 \leftarrow \text{InitializePrimalWeight}(c, q)$;
repeat
    $t \leftarrow 0$;
    repeat
        $z^{n,t+1}, \eta^{n,t+1}, \hat{\eta}^{n,t+1} \leftarrow \text{AdaptiveStepOfPDHG}(z^{n,t}, \omega^n, \hat{\eta}^{n,t}, k)$;
        $\bar{z}^{n,t+1} \leftarrow \frac{1}{\sum_{i=1}^{t+1} \eta^{n,i}} \sum_{i=1}^{t+1} \eta^{n,i} z^{n,i}$;
        $z_c^{n,t+1} \leftarrow \text{GetRestartCandidate}(z^{n,t+1}, z_m^{n,t+1}, z^{n,0})$;
        $t \leftarrow t+1$, $k \leftarrow k+1$;
    until restart or termination criteria holds;
    restart the outer loop $z^{n+1,0} \leftarrow z_c^{n,t}$, $n \leftarrow n+1$;
    $\omega^n \leftarrow \text{PrimalWeightUpdate}(z^{n,0}, z^{n-1,0}, \omega^{n-1})$;
until termination criteria holds;
Output: $z^{n,0}$.

Georgia Institute of Technology

cuPDLP

Adaptive restart based on KKT error

cuPDLP uses the KKT error defined below as a proxy to the normalized duality gap for restarting since KKT error is more computational GPU-friendly.

$$KKT_\omega(z) = \sqrt{d^2 + \omega^2 \left\| \begin{pmatrix} Ax - b \\ (h - Gx)^+ \end{pmatrix} \right\|_2^2 + \frac{1}{\omega^2} \left\| c - K^T y - \lambda \right\|_2^2}$$

$$d = q^T y + l^T \lambda^+ - u^T \lambda^- - c^T x$$

Georgia Institute of Technology

cuPDLP

KKT error restart criteria

1. **Sufficient decay in KKT error**:

$$\mathrm{KKT}_{\omega^n}(z_c^{n,t+1}) \leq \beta_{sufficient}\mathrm{KKT}_{\omega^n}(z^{n,0})$$

2. **Necessary decay + no local progress in KKT error**:

$$\mathrm{KKT}_{\omega^n}(z_c^{n,t+1}) \leq \beta_{necessary}\mathrm{KKT}_{\omega^n}(z^{n,0})$$

$$\mathrm{KKT}_{\omega^n}(z_c^{n,t+1}) > \mathrm{KKT}_{\omega^n}(z_c^{n,t})$$

.

3. **Long inner loop**: $t \geq \beta_{artifical}\, k$.

Georgia Institute of Technology

cuPDLP

Performance on selected large-scale problems

Source	Instance	m	n	nnz	COPT	cuPDLP-C
Koch et al. [8]	zib03	19,731,970	29,128,799	104,422,573	16.5 (h)	916
Pagerank	rand_1m_nodes	1,000,001	1,000,000	7,999,982	-	3.56
	rand_10m_nodes	10,000,001	10,000,000	79,999,982	-	44.22
	com-livejournal	3,997,963	3,997,962	77,358,302	-	21.07
	soc-livejournal1	4,847,572	4,847,571	78,170,533	-	22.26
Unit Com.	ds1	641,037	659,145	21,577,566	592	81
	ds2	641,037	659,145	21,577,566	606	108
Supply-chain	inv-10	4,035,449	3,758,458	15,264,380	-	1636
	inv-20	8,368,795	7,810,584	31,718,673	-	1157
	inv-40	13,186,756	12,066,105	49,528,729	-	6032
	inv-60	16,227,780	14,544,689	60,372,404	-	11102
QAP [1]	wil50	3,437,600	6,252,500	19,125,000	-	96
	lipa50a				-	71
	lipa50b				-	66
	tai50a				-	64
	tai50b				-	437

Figure: Performance on selected large-scale problems

Georgia Institute of Technology

cuPDLP

Partial results of Mittelmann feasible point instances

Method	COPT	cuPDLP-C (COPT)		cuPDLP-C (CLP)		cuPDLP-C	
	10^{-8}	10^{-4}	10^{-8}	10^{-4}	10^{-8}	10^{-4}	10^{-8}
L1_sixm250obs	0.79	6.87	7.99	3.27	5.29	5.34	6.59
Linf_520c	3.07	1253.58	3600.00	11.07	3600.00	13.29	3600.00
a2864-99blp	0.03	0.26	0.34	0.35	0.52	0.53	0.92
bdry2	6.21	3600.00	3600.00	3600.00	3600.00	3600.00	3600.00
cont1	0.09	238.41	602.34	289.77	1337.23	258.78	621.90
cont11	0.19	386.36	3419.51	227.04	3600.00	497.35	2352.79
datt256	0.79	0.17	0.32	0.12	0.24	0.12	0.28
dlr1	26.58	13.36	3600.00	408.29	3600.00	3600.00	3600.00
ex10	0.61	0.06	0.09	0.06	0.10	0.07	0.15
fhnw-binschedule1	21.60	17.50	20.68	55.64	114.75	70.22	75.67
fome13	0.56	1.41	22.23	3.74	11.67	1.54	7.23
graph40-40-1rand	0.02	0.02	0.02	0.05	0.12	0.04	0.09
irish-electricity	11.41	11.63	263.26	5.79	250.40	11.31	3600.00
neos	6.70	22.93	245.26	21.13	212.24	32.58	331.62
neos3	0.11	0.26	0.49	0.39	0.82	0.39	0.80
neos-3025225-shelon	8.51	6.35	6.63	2.79	2.94	2.83	2.97
neos-5052403-cygnet	3.64	9.13	22.56	2.29	9.17	2.21	10.43
neos-5251015-ogosta	1.75	0.73	1.06	0.49	1.00	0.25	2.27
ns1687037	3.02	3600.00	3600.00	237.89	3600.00	3600.00	3600.00
ns1688926	6.55	3600.00	3600.00	3600.00	3600.00	3600.00	3600.00

Figure: Partial results of Mittelmann feasible point instances

cuPDLP

Overall performance (SGM10) on different datasets

Dataset	Optimizer	Presolver	Tol.	SGM10	Solved
MIPLIB (383)	COPT	-	10^{-8}	3.11	383
	cuPDLP-C	COPT	10^{-4}	5.43	379
			10^{-8}	18.53	369
		CLP	10^{-4}	7.95	372
			10^{-8}	21.89	362
		No Presolve	10^{-4}	10.28	370
			10^{-8}	27.15	359
	cuPDLP.jl	No Presolve	10^{-4}	17.49	370
			10^{-8}	35.69	355
Mittelmann (49)	COPT	-	10^{-8}	13.81	48
	cuPDLP-C	COPT	10^{-4}	25.29	46
			10^{-8}	110.22	41
		CLP	10^{-4}	33.97	45
			10^{-8}	125.95	38
		No Presolve	10^{-4}	57.54	43
			10^{-8}	172.98	39

Figure: Overall performance (SGM10) on different datasets

Shifted geometric mean: $SGM := (\Pi_{i=1}^{n}(t_i + \Delta))^{1/n} - \Delta$, where t_i is the solving time for the ith instance and $\Delta = 10$.