

Precomputation-Based Algorithms for Small Multi-Scalar Multiplication over Fixed Bases

Doktorantas: Saulius Grigaitis

Disertacijos pavadinimas: Fiksuotos bazės mažos daugiaskaliarinės daugybos algoritmai, pagrįsti išankstiniais skaičiavimais

Studijų laikas: 2018 – 2026 (akademinės atostogos 2021-2026)

Vadovas: prof. dr. Igoris Belovas

Tyrimo objektas:

Fiksuotos bazės daugiaskaliarinės daugybos algoritmai mažiems uždaviniams, kurie našumui pagerinti naudoja išankstinius skaičiavimus.

Tyrimo tikslas:

Ištirti krepšelių metodo efektyvumą mažai daugiaskaliarinei daugybai, kai taikomi išankstiniai skaičiavimai, ir pasiūlyti šio metodo našias modifikacijas.

Rezultatai

- sukurti nauji efektyvūs išankstiniais skaičiavimais pagrįsti Pippengerio algoritmo variantai;
- įgyvendinti algoritmai vieningoje programinėje sistemoje, palaikančioje kelias ECC bibliotekas, kad jų vertinimas būtų tikslus ir atkuriamas;
- ištirtas siūlomų variantų našumas, palyginant juos su kitais algoritmais.

Plano vykdymo suvestinė

Studijų metai	Egzaminai	
	Planas	Ivykdyta
I (2018/2019)	1	1
II (2019/2020)	2	3
III (2020/2021)	1	
IV (2025/2026)		
Iš viso:	4	4

Studijų metai	Dalyvavimas konferencijose				Publikacijos					
	Tarptautinėse		Nacionalinėse		Su citav. rodikliu			Be citav. rodiklio		
	Planas	Ivykdyta	Planas	Ivykdyta	Planas	Ivykdyta	Būklė	Planas	Ivykdyta	Būklė
I (2018/2019)			1	1 ³						
II (2019/2020)	1					1 ⁶	Publikuota			
III (2020/2021)	1	1 ⁴			1	1 ⁷	Publikuota		1 ²	Publikuota
IV (2025/2026)		2 ^{5,8}			1	2 ^{9,10}	Iteikta			
Iš Viso	2	3	1	1	2	4	4	0	1	1

Mokslinių tyrimų etapai

Darbo pavadinimas		Atlikimo terminai	Pastabos
1.	Mokslinių tyrimų disertacijos tema apžvalga ir analizė (Lietuvoje ir užsienyje): 1.1. Atlikti blokų grandinių tinklų analitinę apžvalgą. 1.2 Nustatyti (identifikuoti) mokslines problemas, kylančias uždaviniuose, susijusiuose su blokų grandinių spartinimu naudojant negrandines (angl. <i>off-chain</i>) transakcijas.	2018 m. spalio mėn. – 2019 m. spalio mėn.	Atspausdinta apžvalginė publikacija šios dalies pagrindu.
2.	Mokslinio tyrimo vykdymas: 2.1. Tyrimo metodikos sudarymas: 2.1.1. Tyrimo metodikos išsikeltam uždaviniui spręsti parinkimas; 2.1.2. Teorinio ir empirinio tyrimų suplanavimas pagal pasirinktą metodiką. 2.2. Teorinis tyrimas: 2.2.1. Sričių, kuriose tikslinga spartinti blokų grandines negrandininėmis transakcijomis identifikavimas; 2.2.2. Blokų grandinių spartinimo naudojant negrandines transakcijas tyrimas; 2.2.3. Blokų grandinių spartinimo naudojant negrandines transakcijas modelio sukūrimas ar testavimas.	2019 m. lapkričio mėn. – 2020 m. sausio mėn. 2020 m. vasario mėn. – 2020 m. spalio mėn.	Fokusuojamasi į simulatorių tyrimus, kurie leistų įvertinti protokolų patobulinimus našumo atžvilgiu. Nustatyta, kad tinkamų simulatorių naujos kartos PoS protokolams nėra.

Mokslinių tyrimų etapai

	2.3. Empirinis tyrimas: 2.3.1. Blokų grandinių spartinimo naudojant negrandinės transakcijas pritaikymas 2.2.1 uždavinyje identifikuotoms praktinėms sritims. 2.3.2. El. komercijai pritaikyto blokų grandinių sprendimo, naudojančio negrandinės transakcijas, tyrimas ir tobulinimas.	2020 m. lapkričio mėn. – 2021 m. gegužės mėn.	Atspausdinta publikacija šios dalies pagrindu. Tolesni tyrimai fokusuojasi į optimizavimą kriptografijos algoritmų (KZG10 ir kt.), kurie įgalina didinti transakcijų našumą duomenų shardingo pagalba
	2.4. Gautų rezultatų analizė, apibendrinimas, išvadų parengimas: 2.4.1. Gautų rezultatų analizė; 2.4.2. Rezultatų apibendrinimas, esminių rezultatų išskyrimas; 2.4.3. Išvadų parengimas.	2021 m. birželio mėn. – 2021 m. spalio mėn.	
3.	Atskirų daktaro disertacijos dalių (tyrimo metodikos, rezultatų, ginamų teiginių, išvadų ir kt.) parengimas: 3.1. Tikslų, uždavinių, tyrimo metodikos, ginamųjų teiginių patikslinimas; 3.2. Analitinės disertacijos dalies parengimas; 3.3. Teorinės disertacijos dalies parengimas; 3.4. Eksperimentinės disertacijos dalies parengimas; 3.5. Bendrųjų išvadų formulavimas.	2021 m. lapkričio mėn. – 2022 m. gegužės mėn.	Planas nusikėlė 4 metams dėl akademinų atostogų į 2025-2026. Pasiūlyta nauja šeima algoritmų konkrečiai mokslinei problemai - multi-skaliarinei daugybai, kuri naudojama tiek ir KZG10

Mokslinių tyrimų etapai

			<u>naudojama tiek ir KZG10</u> schemoje, tiek ir labai plačiai kitur (pvz. parašų verifikavime). <u>Parengta 60%</u> disertacijos juodraščio.
4.	Daktaro disertacijos parengimas ir svarstymas padalinyje	2022 m. birželio mėn. (planas <u>nusikėlė į 2026</u>)	<u>Parengta 90%</u> disertacijos juodraščio.
5.	Daktaro disertacijos gynimas	2022 m. rugsėjo mėn. (planas <u>nusikėlė į 2026</u>)	<u>Parengta 100%</u> disertacijos juodraščio.

Like darbai

Sulaukti įteiktų straipsnių (*CA WoS* žurnaluose su citavimo rodikliu) recenzijų ir pakoreguoti straipsnius pagal recenzentų pastabas.

KZG10 Polynomial Commitments: Trusted Setup

A common (not desired) requirement for Zero Knowledge schemes;

KZG10 needs prover and verifier to know $[s^i]_1$ and $[s^i]_2$ for $i = 0, \dots, n - 1$;

Secret s must be not be known to anybody and it's computation too hard to recover it from the given trusted setup;

Secure Multiparty Computation (MPC) is a secure way to generate Trusted Setup if at least single party is honest;

KZG10 Polynomial Commitments:

Commitment

KZG10 polynomial commitment is a polynomial evaluated at secret s that nobody knows:

$$[p(s)]_1 = [\sum_{i=0}^n p_i s^i]_1 = \sum_{i=0}^n p_i [s^i]_1 \quad (2)$$

The probability that any another polynomial has the same commitment at unknown secret s equals to the degree of polynomial (not higher than trusted setup size) divided by elliptic curve group order. Let's say for the trusted setup of size 2^{26} and elliptic curve order 2^{256} the probability is $2^{26}/2^{256} = 2^{-230}$.

Multi-Scalar Multiplication: Pippenger Example

$$S_{n,r} = \sum_{i=1}^n a_i P_i$$

When r is small we group elements by scalar:

$$S_{12,4} = 2P_1 + 3P_2 + 3P_3 + 2P_4 + 1P_5 + 1P_6 + 3P_7 + 2P_8 + 2P_9 + 3P_{10} + 1P_{11} + 3P_{12} - 37 \text{ group operations}$$

$$S_{12,4} = 1 \cdot (P_5 + P_6 + P_{11}) + 2 \cdot (P_1 + P_4 + P_8 + P_9) + 3 \cdot (P_2 + P_3 + P_7 + P_{10} + P_{12}) = 1S_1 + 2S_2 + 3S_3.$$

$1S_1 + 2S_2 + 3S_3 = S_3 + (S_3 + S_2) + (S_3 + S_2 + S_1)$ – this needs only 4 group operations + 9 group operations for each subsum.

Multi-Scalar Multiplication: Pippenger (large scalars)

When r is large we break down each scalar:

$$a_i = \sum_{j=0}^{h-1} a_{ij} q^j, \quad 0 \leq a_{ij} < q, \quad h = \lceil \log_q r \rceil$$

$$S_j := a_{1,j}P_1 + a_{2,j}P_2 + \cdots + a_{n,j}P_n \quad (0 \leq j \leq h-1)$$

$$\begin{aligned} S_{n,r} &= \sum_{i=1}^n a_i P_i = \sum_{i=1}^n \left(\sum_{j=0}^{h-1} a_{ij} q^j \right) P_i = \sum_{j=0}^{h-1} q^j \left(\sum_{i=1}^n a_{ij} P_i \right) \\ &= \sum_{j=0}^{h-1} q^j S_j, \end{aligned}$$

Multi-Scalar Multiplication: BGMW

Points are fixed for KZG10 (and many other applications of MSM) so we can precompute $q^j P_i$

$$a_i = \sum_{j=0}^{h-1} a_{ij} q^j, \quad 0 \leq a_{ij} < q, \quad h = \lceil \log_q r \rceil$$

$$S_j := a_{1,j} P_1 + a_{2,j} P_2 + \cdots + a_{n,j} P_n \quad (0 \leq j \leq h-1)$$

$$S_{n,r} = \sum_{i=1}^n a_i P_i = \sum_{i=1}^n \left(\sum_{j=0}^{h-1} a_{ij} q^j \right) P_i = \sum_{i=1}^n \sum_{j=0}^{h-1} a_{ij} \cdot q^j P_i$$

This requires at most $nh + q - 3$ group operations.

Extension I

Precompute pairs $P_i + P_j$ (in addition to BGMW) for indices

$$(i, j) \in \bigcup_{d=0}^{\frac{k}{tw}-1} \{(a, b) : a, b \in \{dtn, dtn + 1, \dots, (d+1)tn - 1\}, a < b, \}$$

where w denotes the window size in bits, n is the MSM size, k is the scalar size in bits, and t is the chunk size

Extension II

- Extension I + Booth indexes
- Precompute pairs $P_i + P_j$ and $P_i - P_j$ (in addition to BGMW) for indices

$$(i, j) \in \bigcup_{d=0}^{\frac{k}{tw}-1} \{(a, b) : a, b \in \{dtn, dtn + 1, \dots, (d + 1)tn - 1\}, a < b\}.$$

where w denotes the window size in bits, n is the MSM size, k is the scalar size in bits, and t is the chunk size

Extension III

- Extension II + further buckets reduction
- Precompute pairs $aP_i \pm bP_j$ (in addition to BGMW), where

$$a, b \in \Theta, \quad \Theta = \left\{ 2^\theta \mid \theta \in [0, \dots, w-1] \right\},$$

$$(i, j) \in \bigcup_{d=0}^{\frac{k}{tw}-1} \{(a, b) : a, b \in \{dtn, dtn+1, \dots, (d+1)tn-1\}, a < b\}.$$

where w denotes the window size in bits, n is the MSM size, k is the scalar size in bits, and t is the chunk size

Extension IV

- Extension I + triples.
- Precompute:

$$P_i + P_j,$$

where

$$(i, j) \in \bigcup_{d=0}^{\frac{k}{tw}-1} \{(a, b) : a, b \in \{dtn, dtn + 1, \dots, (d + 1)tn - 1\}, a < b\}$$

and triplet sums

$$P_i + P_j + P_m,$$

where

$$(i, j, m) \in \bigcup_{d=0}^{\frac{k}{tw}-1} \{(a, b, c) : a, b, c \in \{dtn, dtn + 1, \dots, (d + 1)tn - 1\}, a < b < c\}.$$

where w denotes the window size in bits, n is the MSM size, k is the scalar size in bits, and t is the chunk size

Extension V

- Extension IV + Booth encoding.
- Precompute:

$$P_i + P_j, \quad P_i - P_j,$$

where

$$(i, j) \in \bigcup_{d=0}^{\frac{k}{tw}-1} \{(a, b) : a, b \in \{dtn, dtn + 1, \dots, (d + 1)tn - 1\}, a < b\}.$$

and triplet precomputations

$$\begin{array}{ll} P_i + P_j + P_m, & -P_i + P_j + P_m, \\ P_i - P_j + P_m, & P_i + P_j - P_m, \end{array}$$

where

$$(i, j, m) \in \bigcup_{d=0}^{\frac{k}{tw}-1} \{(a, b, c) : a, b, c \in \{dtn, dtn + 1, \dots, (d + 1)tn - 1\}, a < b < c\}.$$

where w denotes the window size in bits, n is the MSM size, k is the scalar size in bits, and t is the chunk size

Extension VI

- Extension V + overlapping chunks.
- Precompute:

$$(i, j) \in \bigcup_{d=0}^{D-1} \{(a, b) : a, b \in \mathcal{A}_{d,\rho}(n, t), a < b\}$$

and

$$(i, j, m) \in \bigcup_{d=0}^{D-1} \{(a, b, c) : a, b, c \in \mathcal{A}_{d,\rho}(n, t), a < b < c\},$$

where

$$\mathcal{A}_{d,\rho}(n, t) = \{d(1 - \rho)tn, d(1 - \rho)tn + 1, \dots, d(1 - \rho)tn + tn - 1\}$$

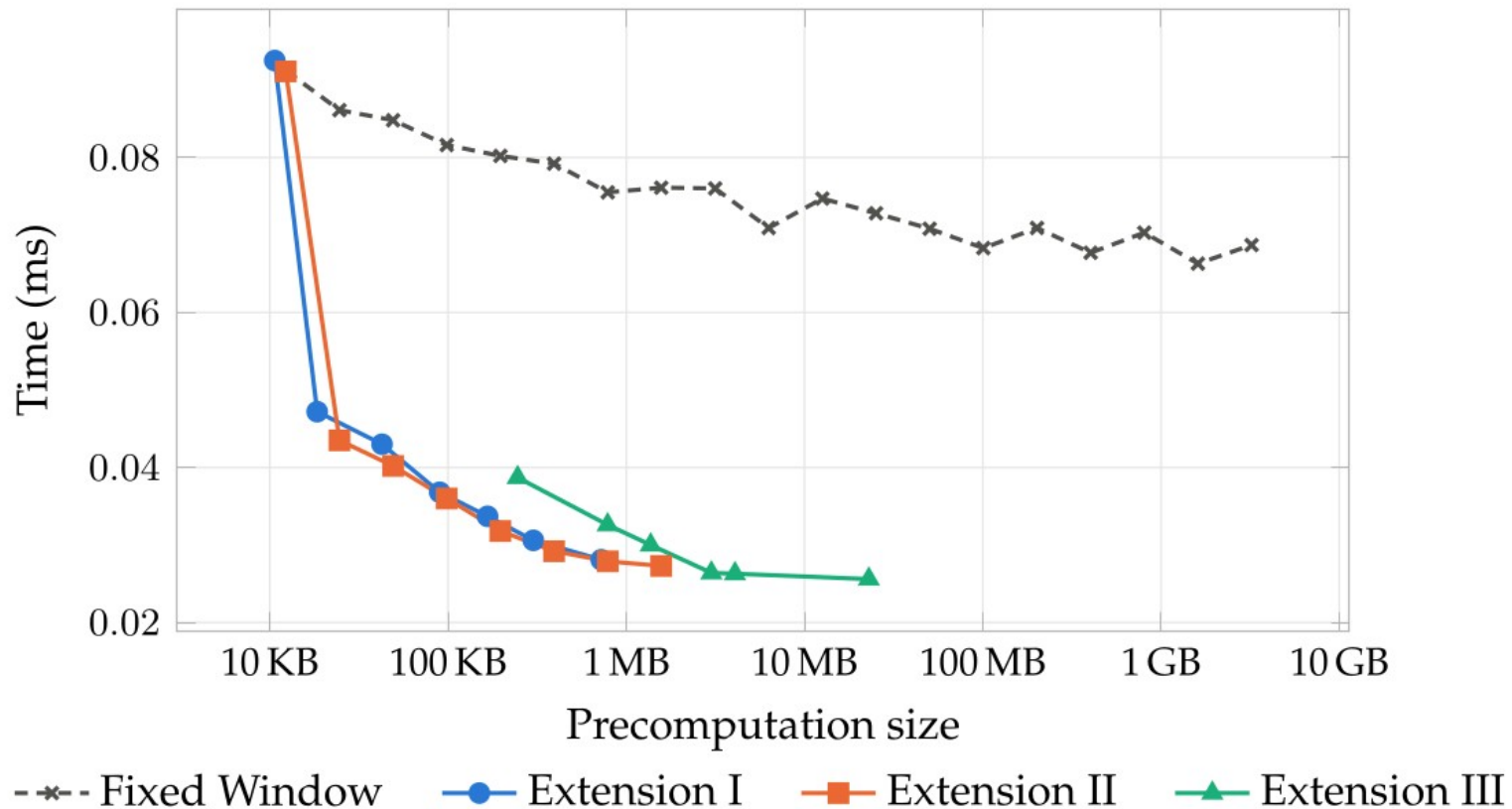
and the number of overlapping chunks D is given by

$$D = \left\lceil \frac{h - t}{(1 - \rho)t} \right\rceil + 1.$$

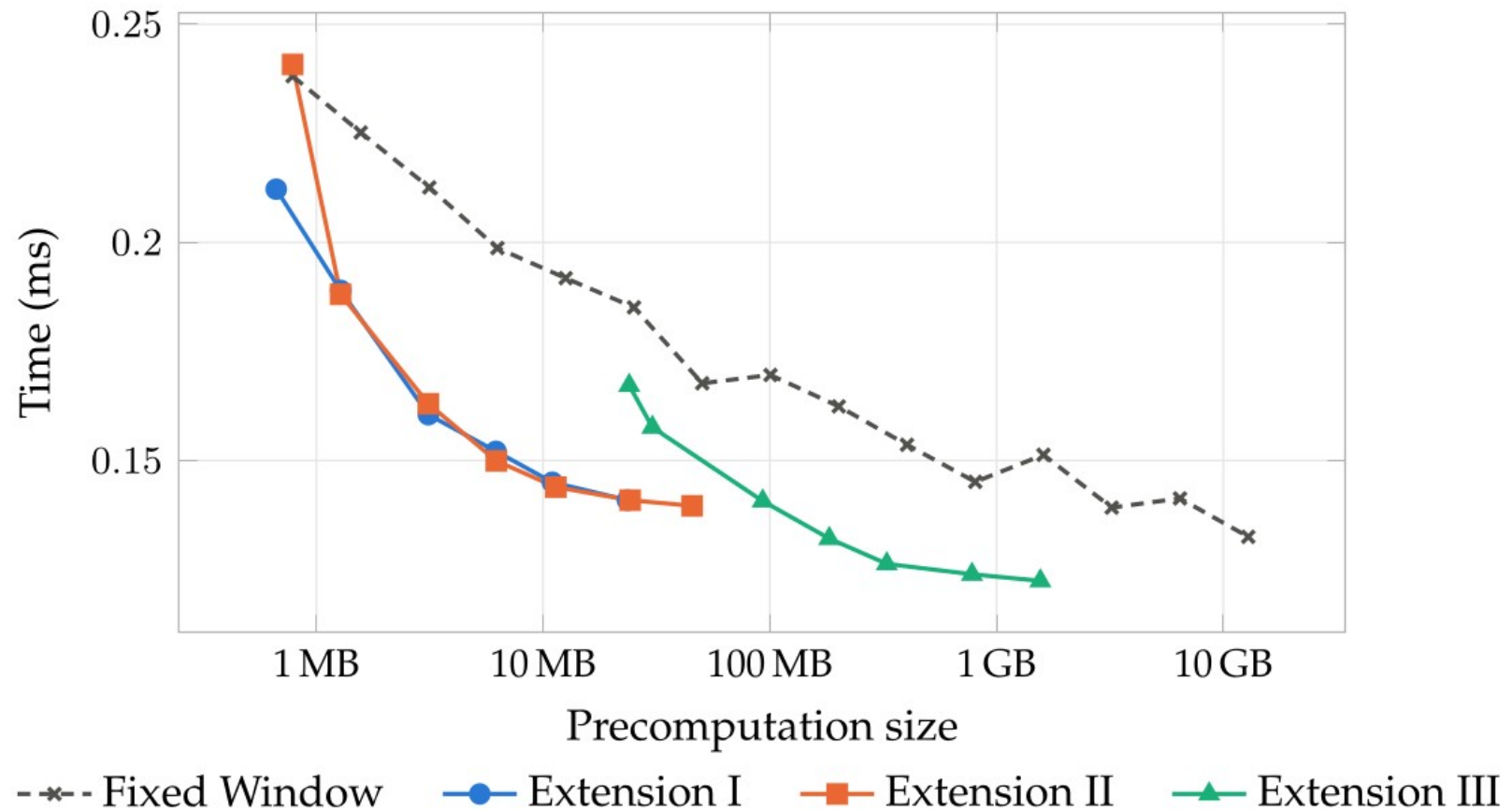
Experiments: Rust-kzg library

- Implemented in Rust (high performance and safety);
- Contains test suites and benchmarks;
- Used by some real world projects and scientists;
- Over 40 students involved;
- Opensource (Apache 2.0 license).

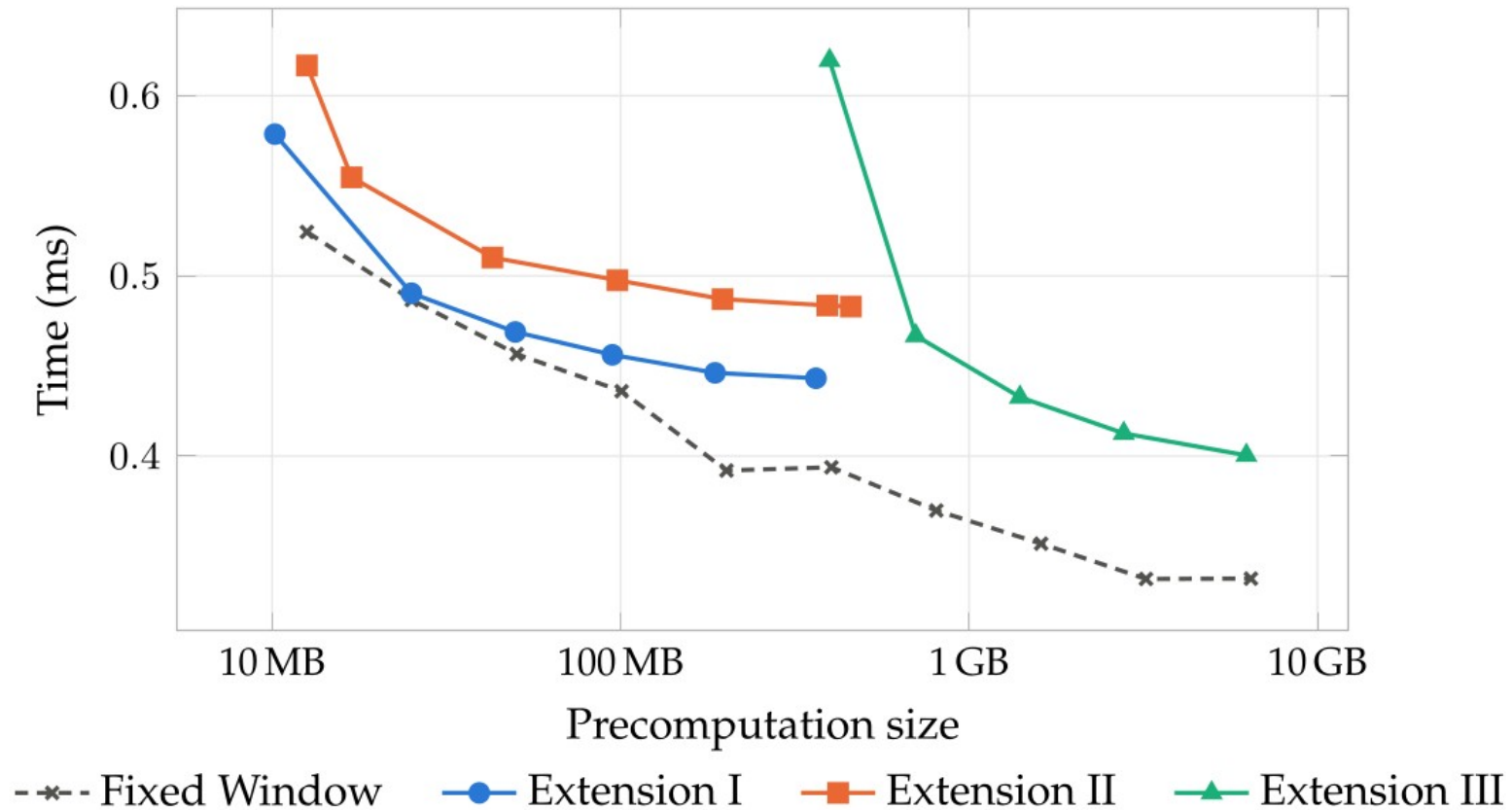
MSM size 2^1 : Fixed Window vs. Extensions I–III



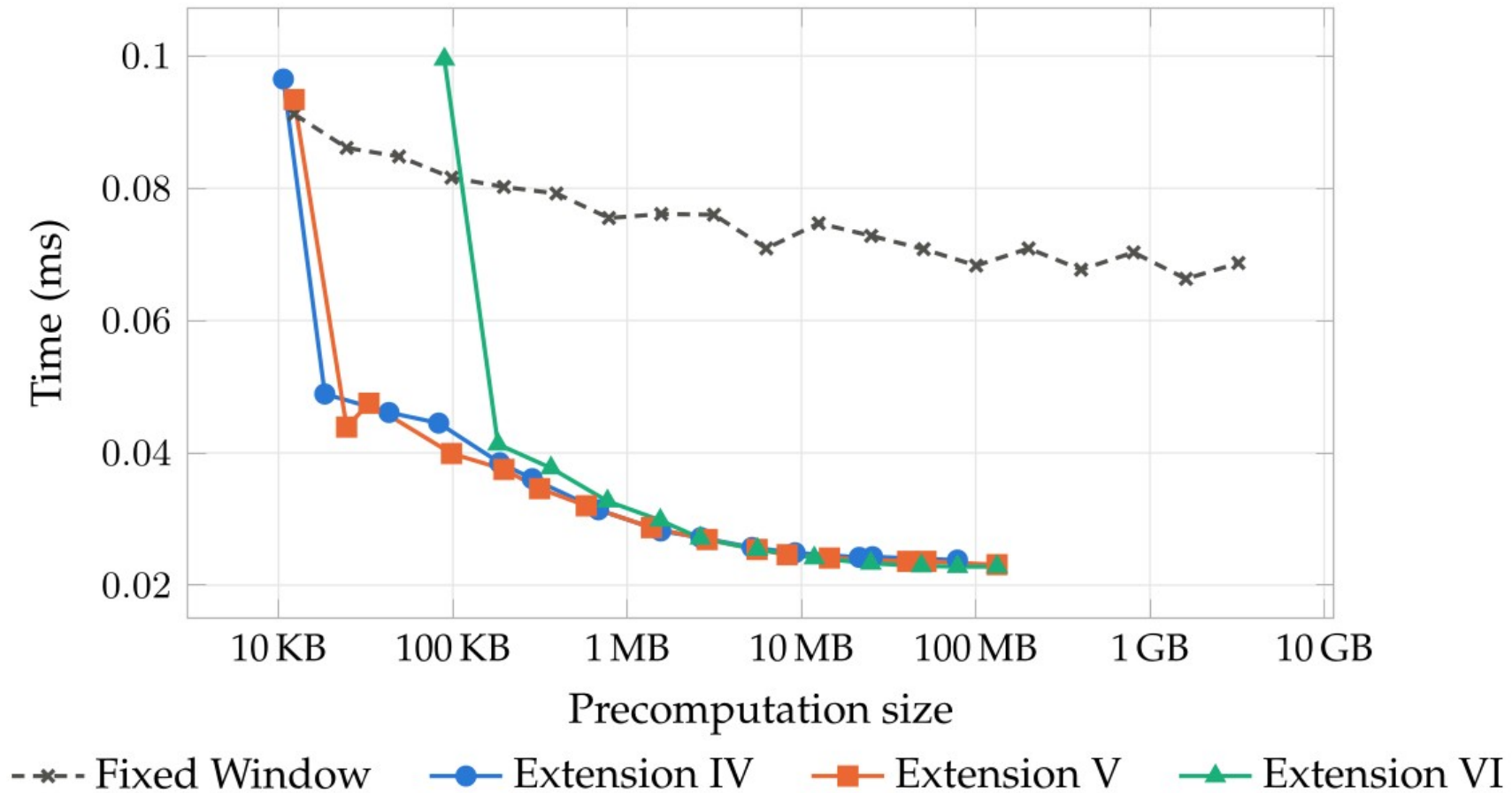
MSM size 2^4 : Fixed Window vs. Extensions I–III



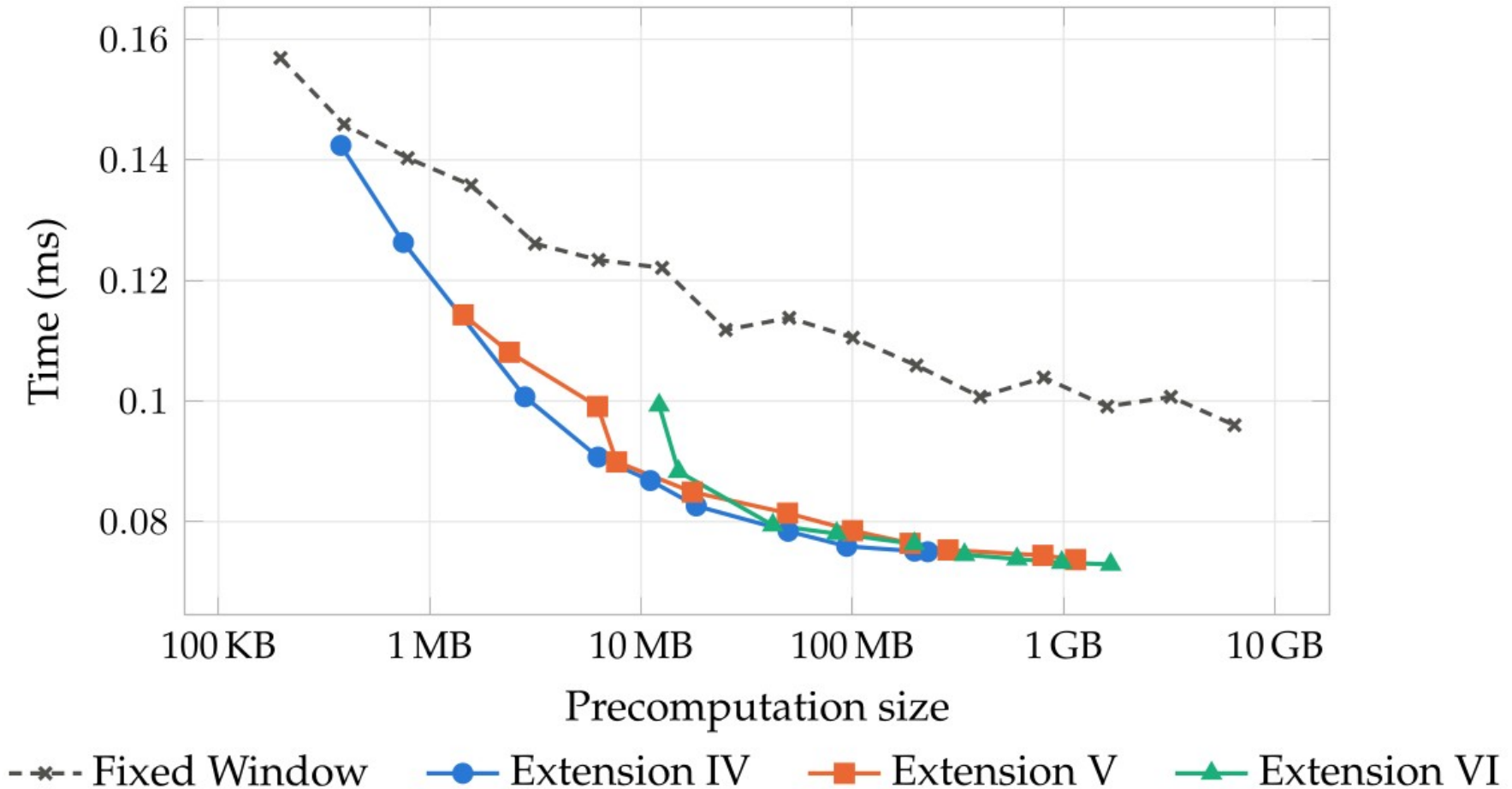
MSM size 2^6 : Fixed Window vs. Extensions I–III



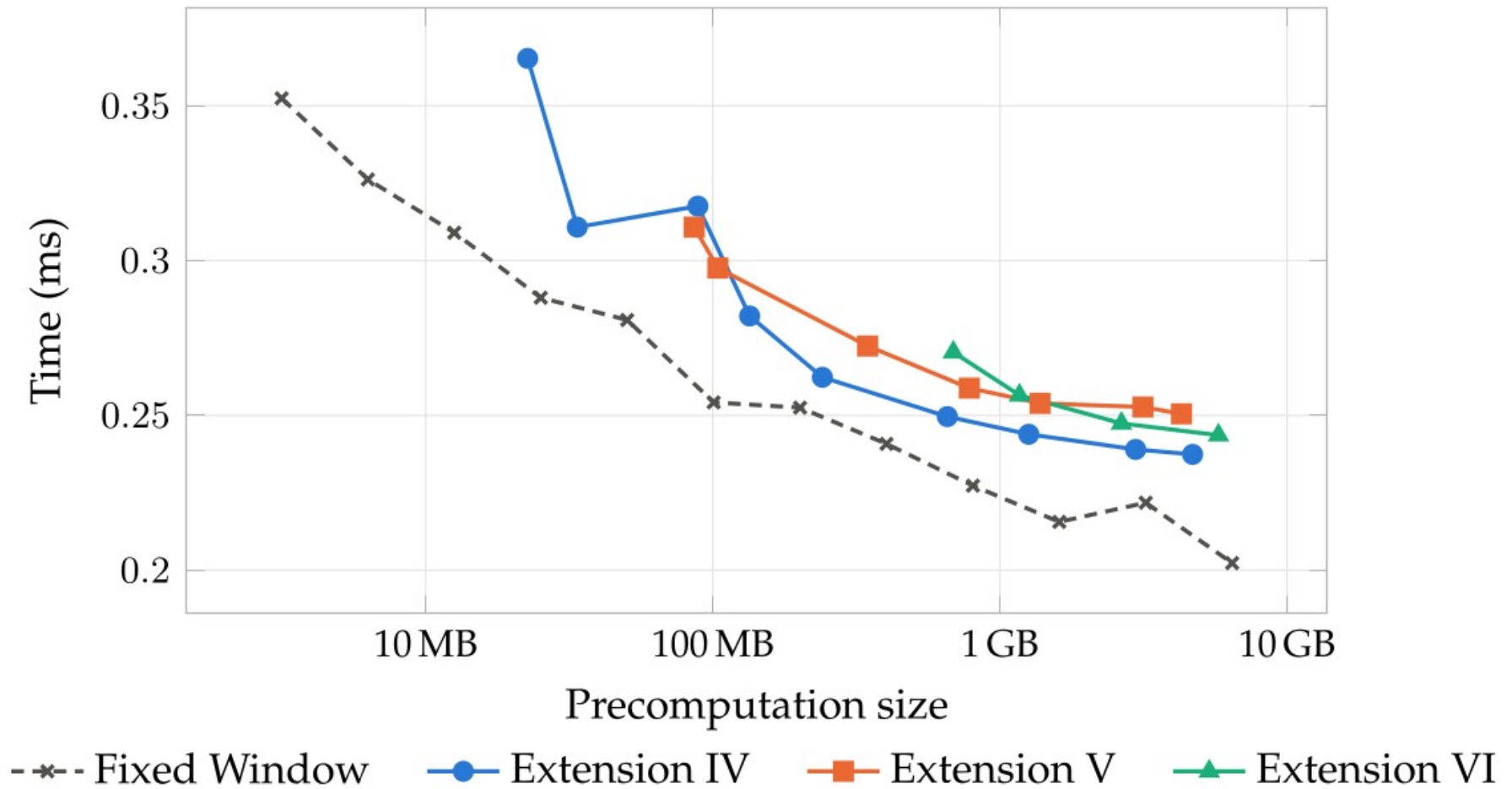
MSM size 2^1 : Fixed Window vs. Extensions IV–VI



MSM size 2^3 : Fixed Window vs. Extensions IV–VI



MSM size 2^5 : Fixed Window vs. Extensions IV–VI



Questions