Erhaltene Teilchengruppen und parallele Bucket-Partitionierung

Pascal Bauer

Entwicklung und Auswertung mit ChatGPT

18. September 2026

1 Fragestellung und Stand

Pascal Bauer schlug vor, die raeumliche CFK-Ordnung und rekursive Buckets zur Vorsortierung und Parallelisierung auszunutzen. ChatGPT implementierte zwei experimentelle Varianten und die vorliegende kontrollierte Testserie. Ziel ist die Vermeidung wiederholter globaler Vergleichssortierungen sowie die parallele Bearbeitung eines einzelnen dichten Klumpens. Die Referenz bleibt Standard; die Arbeit Version 0.3 bleibt unveraendert.

Rechner	Prozesse	geplant	Zustaende	Minuten
dreihemden	195	195	456	13.0
schmalzberg	159	159	384	24.6

Status: dreihemden: complete, schmalzberg: complete.

Jeder abgeschlossene Zustand bestand native exakte Vektorvergleiche fuer Knoten, notwendige Teilungen, Teilchenbesitzer und Tiefenhistogramme. Zusaetzlich wurden Gitterkonformitaet, rechneruebergreifende Digests und archivierte Quell-/Binaerpruefsummen geprueft. Diese Aussagen gelten fuer getestete Eingaben, nicht als universeller Korrektheitsbeweis.

2 Vorab festgelegte Kontrollen

Je drei Seeds (827–829), ein vollstaendiger Warm-up pro Prozess, gemischte Variantenreihenfolge innerhalb gleicher Eingaben. 300000 Teilchen in vier Verteilungen bei physischer Kernzahl; Hotspot-Skalierung mit 100000 Teilchen bei jeder Threadzahl bis 12 bzw. 8. Belastung mit einer Million Teilchen und bewegte Folgen mit acht Messzustaenden bei 300000 Teilchen. Keine konkurrierenden Benchmarkprozesse auf demselben Rechner. Native Aufbauzeit ohne Eingabeerzeugung, Referenzkontrollen und Datei-I/O; Prozesslaufzeit und Peak-RSS enthalten diese dagegen.

Beide neuen Pfade verwenden denselben seriellen Konformitaetsabschluss wie die Referenz. Teilungslisten und finale Knoten werden weiterhin sortiert; vermieden wird nur das wiederholte globale Sortieren aktiver Teilchen. Der alte Parallelpfad hat dagegen einen privat parallelen Abschluss samt Merge. Unterschiede zu ihm duerfen nicht allein der Praefixbildung zugeschrieben werden.

3 Algorithmus und analytische Konstanten

Fuer die geordneten Ecken A, B, C gilt $P = (1 - (b + c)/D)A + (b/D)B + (c/D)C$, wobei b, c, D exakte Integer sind. Die bestehende baryzentrische Darstellung liefert die Kindentscheidung $b < c$ fuer rechts; Gleichheit gehoert links. Die beiden fest einprogrammierten Transformationen lauten

$$T_0(b, c) = (D - b - c, b - c), \qquad T_1(b, c) = (c - b, D - b - c).$$

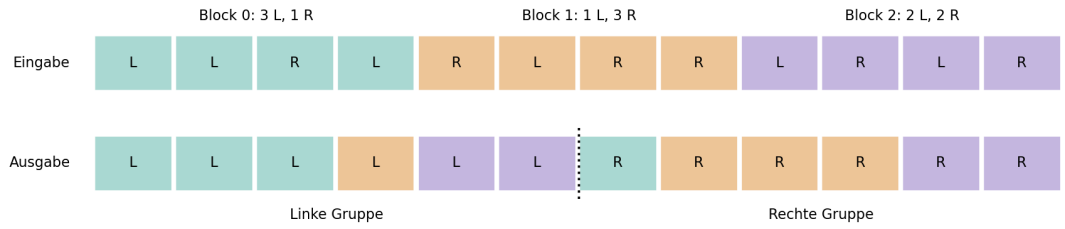
Die dritte baryzentrische Zaehlerkomponente wird dabei links $2c$, rechts $2b$; alle Komponenten bleiben im jeweiligen Kind nichtnegativ und summieren sich zu D. Diese Formeln kodieren die Orientierungswechsel bereits implizit. Keine numerisch geschaetzten Symmetrien und keine allgemeinen Dreiecksoperationen sind noetig. Die 64-Bit-Blattadresse erlaubt maximal 63 Pfadbits; eine kleinere vom Aufrufer gesetzte Grenze wird weiterhin geprueft. Extreme Double-Eingaben verwenden wie die Referenz beliebig grosse Integer statt unsicherer Rundungen.

3.1 Zaehlen, Offsets, Streuen

Ein zusammenhaengender Indexbereich gehoert zu einem CFK-Praefix. Bloecke von hoechstens 2048 Teilchen zaehlen unabhaengig ihre linken Eintraege. Fuer Block k mit Laenge n_k und linker Anzahl l_k sind bei Gruppenbeginn B die Zielanfaenge

$$L_k = B + \sum_{j<k} l_j, \qquad R_k = B + \sum_{j} l_j + \sum_{j<k} (n_j - l_j).$$

Somit sind alle Schreibbereiche disjunkt. Das parallele Streuen transformiert jeden aktiven Punkt genau einmal. Einzelteilchen sind fertig; nur kollidierende Gruppen verbleiben. Indizes werden umgeordnet, die urspruengliche Eingabereihenfolge bleibt fuer die finale Besitzerzuordnung erhalten.

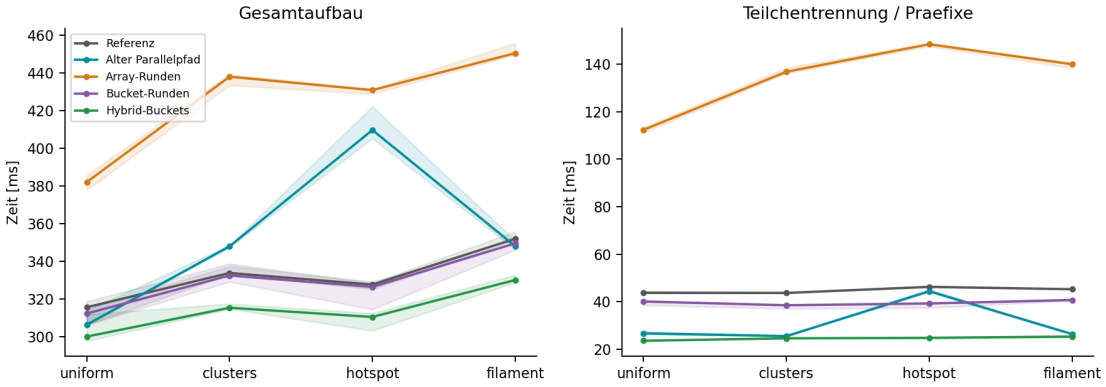


Schematisches Beispiel mit zwoelf Teilchen, keine Messdaten. Farben bezeichnen den Eingabeblock; L/R die Kindentscheidung. Die Ausgabebereiche jedes Blocks sind durch Zaehler und Praefixsummen vorab bestimmt.

3.2 Hybrider Aufbau

Die Kontrollvariante bucket behandelt jede Tiefe in Runden. Die Variante adaptive bearbeitet Gruppen bis 2048 Teilchen dagegen mit lokalem Stack vollstaendig. OpenMP verteilt solche Aufgaben dynamisch. Grosse Gruppen bleiben blockweise parallel, auch wenn fast alle Teilchen in derselben Gruppe liegen. Das ist kein eigener Work-Stealing-Scheduler. Die feste Granularitaet 2048 ist eine technische Startwahl, kein ermitteltes Optimum.

4 Verteilungen: dreihemden

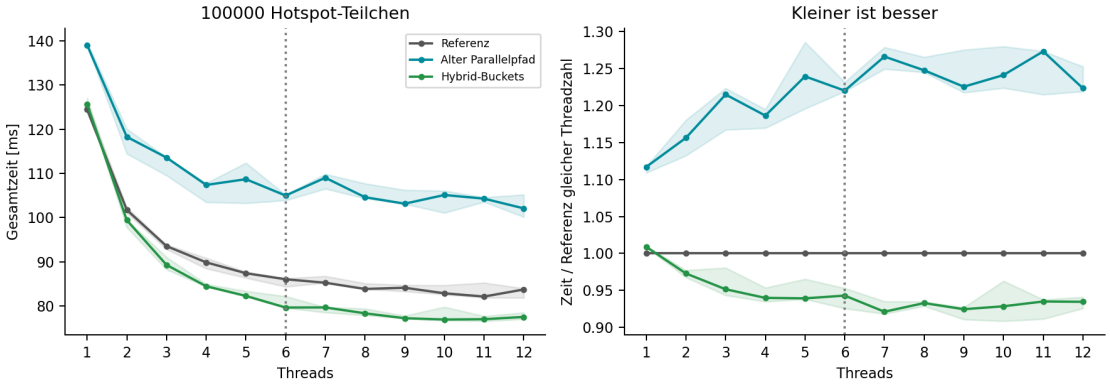


300000 Teilchen, physische Kernzahl. Linien: Median der drei Prozessmediane; Baender: volle Spannweite der drei Seeds. Links Gesamtaufbau, rechts die isolierte Praefixphase. Zwei Frames pro Prozess sind keine unabhaengigen Replikate.

Form	Hybrid/Ref.	Hybrid/Parallel	Praefixe Hybrid/Ref.
uniform	0.980	0.981	0.535
clusters	0.944	0.906	0.563
hotspot	0.951	0.763	0.538
filament	0.936	0.948	0.560

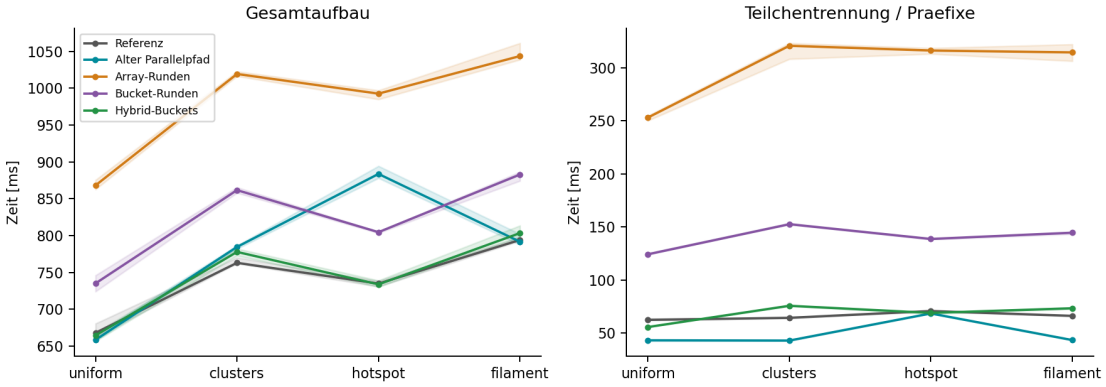
Gepaarte Quotienten je Seed, danach Median. Kleiner als eins bedeutet schneller.

5 Thread-Skalierung: dreihemden



Jede Threadzahl bis zur Hardwarethreadzahl ist enthalten. Senkrechte Linie: physische Kernzahl (6 bzw. 4). Die Referenz ist nur in der Konstruktion seriell; die finale Besitzersuche nutzt ebenfalls die angeforderte Threadzahl. Daher kann auch die Referenzkurve skalieren. Hier gibt es keine Ueberbelegung ueber die Hardwarethreadzahl hinaus.

6 Verteilungen: schmalzberg

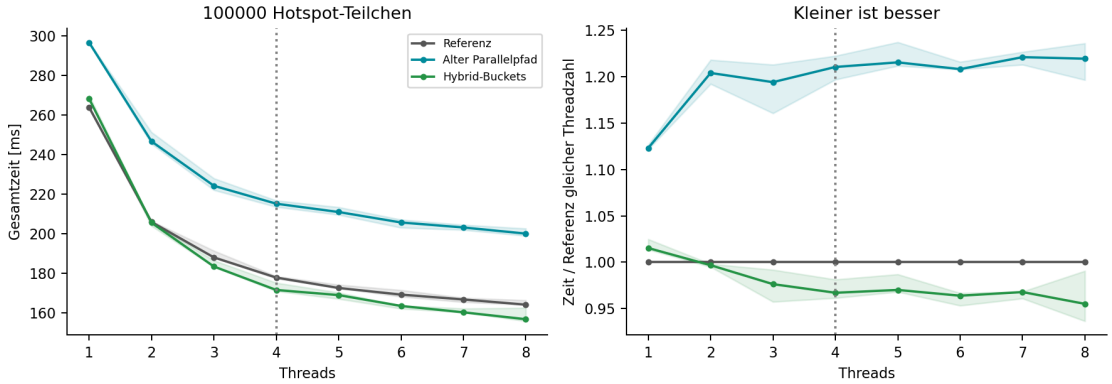


300000 Teilchen, physische Kernzahl. Linien: Median der drei Prozessmediane; Baender: volle Spannweite der drei Seeds. Links Gesamtaufbau, rechts die isolierte Praefixphase. Zwei Frames pro Prozess sind keine unabhaengigen Replikate.

Form	Hybrid/Ref.	Hybrid/Parallel	Praefixe Hybrid/Ref.
uniform	0.983	1.009	0.891
clusters	1.016	0.995	1.177
hotspot	1.000	0.832	0.975
filament	1.007	1.013	1.108

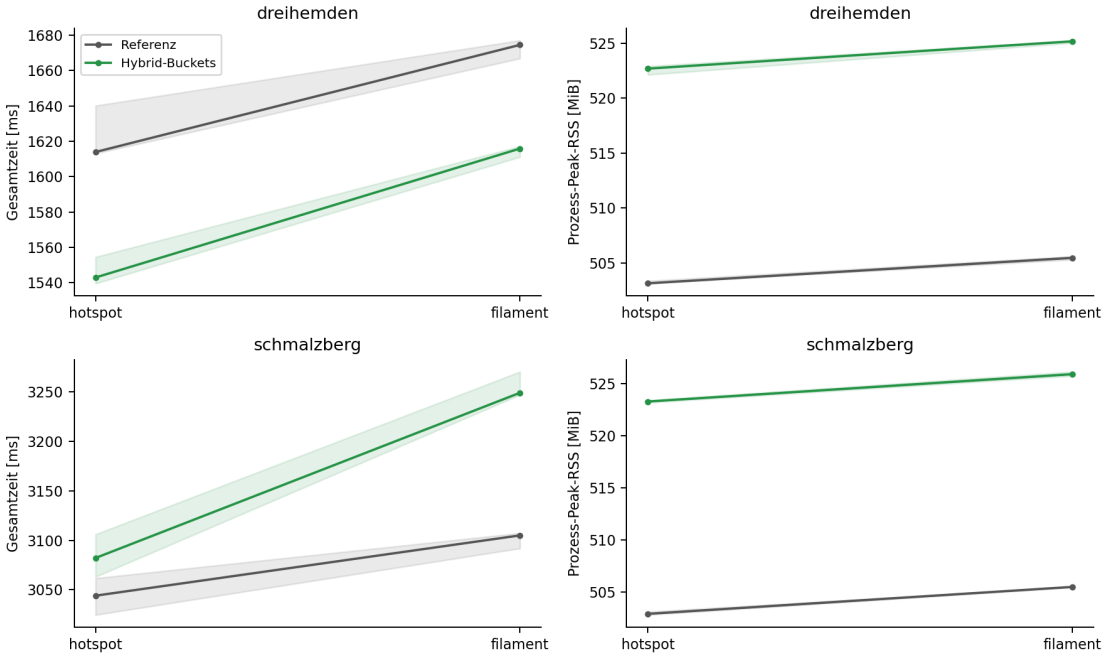
Gepaarte Quotienten je Seed, danach Median. Kleiner als eins bedeutet schneller.

7 Thread-Skalierung: schmalzberg



Jede Threadzahl bis zur Hardwarethreadzahl ist enthalten. Senkrechte Linie: physische Kernzahl (6 bzw. 4). Die Referenz ist nur in der Konstruktion seriell; die finale Besitzersuche nutzt ebenfalls die angeforderte Threadzahl. Daher kann auch die Referenzkurve skalieren. Hier gibt es keine Ueberbelegung ueber die Hardwarethreadzahl hinaus.

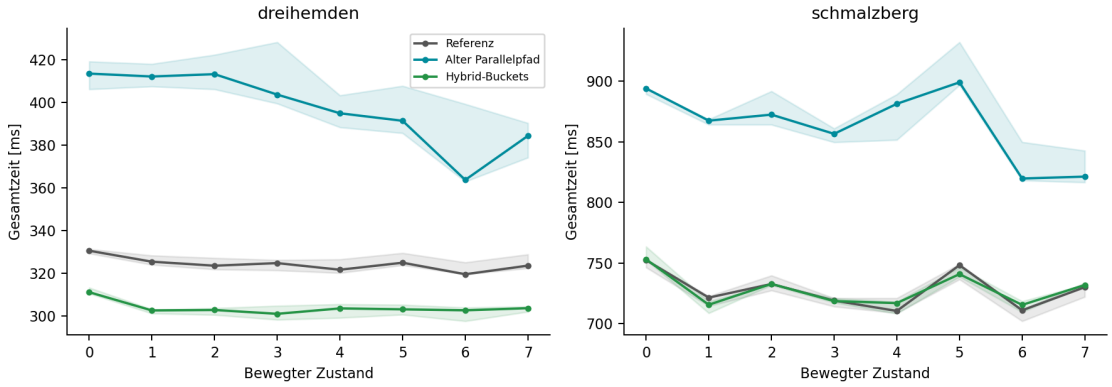
8 Belastung mit einer Million Teilchen



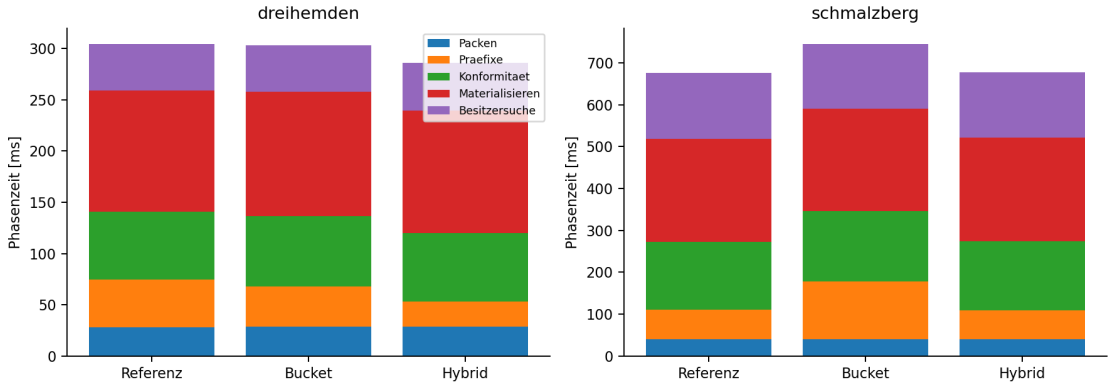
Drei Seeds, je drei gemessene Zustaende. Hotspot: 90 Prozent der Teilchen in einem kleinen Quadrat; Filament: schmaler gewundener Streifen. Rechte Spalte: Prozess-Peak-RSS einschliesslich Referenzbaum und Kontrollen, nicht der isolierte Speicherbedarf eines einzelnen Baumes.

Rechner	Form	Hybrid/Referenz
dreihemden	hotspot	0.954
dreihemden	filament	0.966
schmalzberg	hotspot	1.013
schmalzberg	filament	1.051

9 Bewegung und verbleibende Kosten

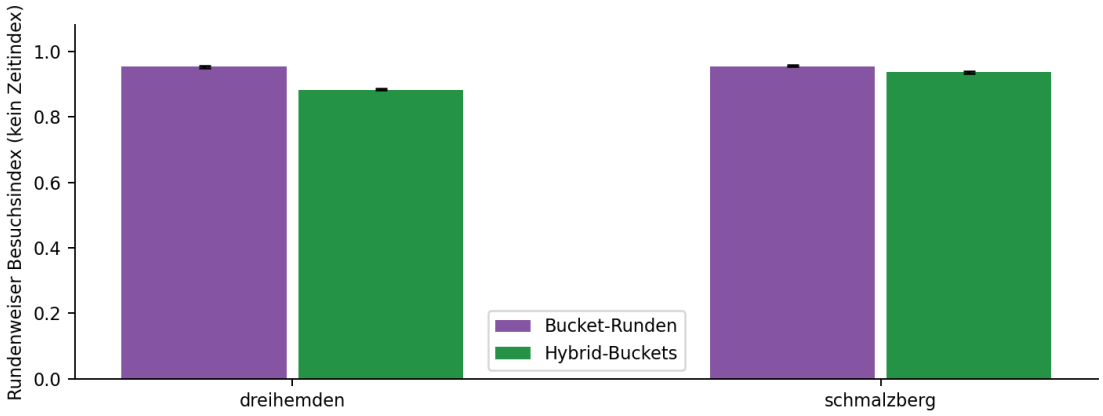


Acht bewegte Zustaende mit 300000 Hotspot-Teilchen. Keine physikalische Zeitintegration: vorgegebene Bewegung prueft wiederholten vollstaendigen Baumaufbau. Eine inkrementelle Wiederverwendung alter Teilchenordnung ist noch nicht implementiert.



300000 Hotspot-Teilchen bei physischer Kernzahl. Median je Hauptphase; Summe von Medianen kann geringfuegig vom Median der Gesamtzeit abweichen. Restzeiten fuer Freigaben sind nicht dargestellt.

10 Lastverteilung: gemessene Arbeit statt Wartezeit



Separate Profilprozesse, 300000 Hotspot-Teilchen, physische Kernzahl. Fuer Besuche v_{rj} in Runde r auf Thread j wird $I = \sum_{rj} v_{rj} / (p \sum_r \max_j v_{rj})$ gebildet. Erst Median ueber Frames, dann ueber drei Seeds; Fehlerbalken zeigen deren Spannweite.

Ein Besuch bedeutet hier eine Kindtransformation; der vorgeschaltete Zaehldurchlauf grosser Gruppen wird nicht mitgezaehlt. Lokale Partitionierung und blockweises Streuen haben unterschiedliche Kosten pro Besuch. Der Index prueft deshalb die Verteilung eines Arbeitsmengen-Proxys, nicht aktive CPU-Zeit oder Beschleunigung. Die rundenweise Berechnung verhindert, dass wechselnde Nachzuegler beim Summieren ueber den ganzen Lauf unsichtbar werden. Innere Barrieren sind in den Threadzeiten enthalten; aus diesen wuerde ein irrefuehrend guter Zeit-Balanceindex entstehen.

11 Bewertung und weitere Arbeit

Die Daten trennen den Nutzen erhaltener Gruppen von den Kosten vollstaendiger Gitterherstellung. Massgebend fuer einen Produktionswechsel ist die Gesamtzeit, nicht allein ein schneller Praefixschritt. Dieser Versuch laesst den Standard unveraendert, auch wenn einzelne Varianten gewinnen.

dreihemden: Hybrid/Referenz ueber die vier Verteilungen bei 300000 Teilchen zwischen 0.936 und 0.980. Bewegte Hotspotfolge: 0.933; gegen alten Parallelpfad 0.749.

schmalzberg: Hybrid/Referenz ueber die vier Verteilungen bei 300000 Teilchen zwischen 0.983 und 1.016. Bewegte Hotspotfolge: 1.000; gegen alten Parallelpfad 0.837.

11.1 Korrektheit und Messgrenzen

Wichtige Einschraenkung: Beim Million-Teilchen-Test auf Schmalzberg ist Hybrid gegenueber der Referenz im Hotspot etwa 1%, im Filament etwa 5% langsamer. Eine deutlich schnellere Variante auf allen Rechnern und Groessen ist damit nicht nachgewiesen. Die einfache Bucket-Rundenvariante ist vor allem eine Kontrolle; erst der lokale Abschluss kleiner Gruppen vermeidet viele teure Runden und Synchronisationen.

Lokale Tests umfassen Randpunkte, Gleichheit auf Trennkanten, leere/einzelne Eingaben, Duplikate, nicht trennbare Punkte, Tiefengrenzen sowie den cpp_int-Fallback. Zusaetzlich wurden der Build ohne OpenMP und Address-/UndefinedBehavior-Sanitizer (ohne Leak-Pruefung) getestet. Kein formaler Race-Freiheitsbeweis und keine ThreadSanitizer-Pruefung. Der Datenbesitz ist disjunkt; die realen Paralleltests vergleichen alle Ergebnisvektoren.

Drei Seeds erlauben eine erste robuste Einordnung, aber keine allgemeine Hardwareprognose. Temperatur und Takt wurden nicht geregelt. Unterschiedliche Compiler auf beiden Rechnern verhindern einen reinen Prozessorvergleich. Profilierte Messungen sind separat: die neuen Prefix-Threadzeiten enthalten innere Barrierewartezeit. Ihr Gleichlauf darf nicht als Beweis perfekten Load-Balancings ausgegeben werden. Thread-Teilchenbesuche und Gesamtzeiten stehen in den Rohdaten; unprofilierte Zeiten bestimmen die Leistungsaussagen.

11.2 Prioritaeten

Zuerst den hybriden Pfad mit den dokumentierten exakten Kontrollen weiter absichern und die Granularitaet unabhaengig variieren. Danach die dominierenden verbleibenden Phasen gezielt angehen: effizientere Konformitaet und Materialisierung, anschliessend Wiederverwendung der Ordnung bei Bewegung. Ein geometrisches Quadrat-Bucket-Verfahren oder ein direkter Pfadschluessel ist ein separater Vergleich, nicht Bestandteil dieses Versuchs. Auch eine Raumkurven-Stetigkeit wurde hier weder benoetigt noch bewiesen.

Archiv: eingefrorener Quellstand, Compilerflags, Binaries, Rohdaten, lokale Tests, Pruefsummen und diese reproduzierbare Auswertung. Keine neue physikalische Simulation wurde durchgefuehrt.

Nach Einfrieren des Messstands wurden im Arbeitscode zwei defensive Details ergaenzt: Vorabreservierung des naechsten Gruppenpuffers ausserhalb der OpenMP-Region und Beschraenkung der Diagnosezaehlersumme auf die reale Teamgroesse. Die lokalen Kontrollen wurden danach wiederholt. Alle Leistungsaussagen beziehen sich ausschliesslich auf den unveraendert archivierten Messstand, nicht auf eine separate Vermessung dieser Ergaenzungen.