УНИВЕРСАЛЕН АГРЕГАТОР $A\circ$ ФОРМАЛНА ТЕОРИЯ, СВОЙСТВА И ПРИЛОЖЕНИЯ

Ангел Тошков
Бургаски свободен университет, e-mail: angel@bfu.bg

Абстракт: *В монорафията е представена формална теория на универсалния агрегатор $A\circ$ – оператор, обединяващ широк клас дискретни редукционни процеси: суми, произведения, екстремуми, булеви квантори, катаморфизми, семирингови редукции, оценки върху графи, езикови операции, композиции на трансформации и други. Въз основа на обща схема за избор, оценяване и структурирано комбиниране чрез бинарна операция, агрегаторът се разглежда като единен апарат, който съчетава класически логически системи [1], категорийни конструкции [5], алгебра на програмирането [11], институции [16] и семирингови модели [19]. Дефинира се разширена версия $A\circ^*$, която не въвежда нов оператор, а надгражда първоначалната дефиниция чрез контекст, зависими оценки, произволни крайни структури и контролирано структуриране на редукцията. Извеждат се основни свойства и теореми, доказващи универсалността на агрегатора като общ редукционен оператор за дискретни структури. Показват се примери и приложения в логика, семантика, графови алгоритми и специализирани алгебрични домейни.*

Ключови думи: *агрегиране; универсален оператор; редукция; моноиди; катаморфизми; семиринги; институции; логически квантори; композиции.*

THE UNIVERSAL AGGREGATOR $A\circ$ FORMAL THEORY, PROPERTIES AND APPLICATIONS

Angel Toshkov
Burgas Free University, angel@bfu.bg

Abstract: *This work presents an formal theory of the universal aggregator $A\circ$, an operator that unifies a broad class of discrete reduction processes: sums, products, extrema, Boolean quantifiers, catamorphisms, semiring reductions, graph evaluations, language operations, compositions of transformations and others. Based on a general scheme of selection, evaluation, and structured combination via a binary operation, the aggregator is introduced as a unifying mechanism that incorporates classical logical foundations [1], category–theoretic constructions [5], algebra of programming [11], institutional model theory [16], and semiring-based algebraic systems [19]. An extended definition $A\circ^*$ is provided, not as a new operator but as a generalization of the original $A\circ$, incorporating context, dependent evaluations, arbitrary finite structures, and controlled reduction order. Fundamental properties and theorems are established, demonstrating the aggregator's universality as a general reduction operator over discrete structures. Applications span logic, semantics, graph algorithms, and specialized algebraic domains.*

Keywords: *aggregation; universal operator; reduction; monoids; catamorphisms; semirings; institutions; logic; compositions.*

1. Увод

Агрегирането на дискретни стойности чрез определен ред, условие или алгебрична операция е фундаментален механизъм в математиката, логиката, информатиката и семантиката на формални езици. Класическите оператори като сума, произведение, минимум/максимум, логически квантори ∀ и ∃, редукции в семирингови структури [19], композиции на функции и структурни катаморфизми [11] изглеждат на пръв поглед различни, но споделят общ модел: (1) избор на активни елементи; (2) оценяване чрез функция; (3) комбиниране чрез бинарна операция; (4) структура на редукцията (ред и групиране със скоби); (5) евентуална зависимост от контекст.

Съществуващите изследвания по логика (Клини [1]), по алгебрични структури (Бъркоф–Маклейн [6]), по категорийна теория (Маклейн [5], Лоувер [9]), по институции (Гогуен–Бърстал [16]) и по семирингови модели [19] разглеждат конкретни аспекти на агрегирането, но не в обща рамка. Настоящата монография предлага именно такъв обединен подход.

Във въведената в [28] дефиниция агрегаторът $A \circ$ се основава на комутативна и асоциативна операция и се прилага върху множество елементи. В последващото разширение [29] се анализират свойства като асоциативност, монотонност, инвариантност и неутралност. В настоящата работа тези идеи се доразвиват чрез въвеждане на разширен агрегатор $A \circ^*$, който обобщава всички класически случаи, без да променя същността на оператора $A \circ$.

2. Литературен обзор

Настоящият раздел представя преглед на фундаменталните теоретични направления, които формират основата за изучаването на универсалния агрегатор $A \circ$ и неговото разширение $A \circ^*$. Обзорът включва логически, алгебрични, категорийни, семирингови, програмистки и семантични теории, които оказват значимо влияние върху стандартизирането на понятието „агрегиращ оператор“.

2.1. Логически и метаматематически основи (Kleene, Tarski, Church)

Формалните логически системи, дефинирани от Клини [1], Тарски [3] и Чърч [4], въвеждат основите на предикатна логика, булеви оператори, логически квантори ∀ и ∃, рекурсивни функции, семантики на истинност и др.

Булевите квантори могат да бъдат формализирани като специални случаи на агрегиращи оператори. В частност:

$$\forall x \in X\, P(x) \iff \bigwedge_{x \in X} P(x),$$

$$\exists x \in X\, P(x) \iff \bigvee_{x \in X} P(x).$$

Тук $\wedge$ и $\vee$ са бинарни операции върху булеви стойности. Именно този механизъм показва първата концептуална аналогия между логиката и агрегиращите редукции.

2.2. Категорийни основи (Mac Lane, Lawvere, Barr & Wells)

Категорийната теория, формализирана в класическите трудове на Маклейн [5] и Лоувер [9], дефинира категории C, морфизми $f: A \to B$, функтори $F: C \to D$, естествени преобразувания, моноидни обекти и свободни моноиди, както и техните универсални свойства.

Интерес за настоящото изследване представляват няколко обекта.

2.2.1. Моноидни обекти

Моноидният обект в категория C е троица:

$$(M, \mu, \eta),$$

където:

$\mu: M \otimes M \to M$ — „умножение“,

$\eta: I \to M$ — „единичен“ морфизъм.

Този модел обобщава всяка асоциативна бинарна операция. Агрегаторът $A\circ$ използва точно такава структура върху множество S:

$$(S, \circ, e).$$

2.2.2. Свободни структури

В теорията на категориите свободният комутативен моноид над множество D е:

$$D^*/\sim,$$

което е концептуално аналогично на: множество от крайни multisets над D и редукция чрез комутативна операция.

Това дава точното универсално свойство на базовия оператор $A\circ$.

2.3. Алгебра на програмирането и структурни редукции (Bird & de Moor)

Трудът „Algebra of Programming“ [11] формализира структурни редукции като катаморфизми, анаморфизми, хистоморфизми, параметризуеми fold-ове.

$$\text{fold}_{\circ}(x_1, \dots, x_n) = x_1 \circ \cdots \circ x_n.$$

Структурният оператор „fold“ е еквивалентен на агрегатора $A\circ$ при даден ред (линеен вход) и асоциативна операция.

Разширеният агрегатор $A \circ^*$ включва това като частен случай, като допуска и дървовидни структури X, графови структури, контекстно зависимо $f(x, c)$, локални структури на комбиниране $B(X)$ и други.

2.4. Институции и абстрактна моделна теория (Goguen & Burstall)

Основната идея в институциите [16] е разделението между синтаксис, семантика, моделиране и удовлетворимост.

Институцията $I = (Sign, Sen, Mod, \vDash)$ е четворка, която обобщава логическите системи. Агрегаторът $A\circ$ може да бъде третиран като оператор, който редуцира множество от формули/модели, комбинира локални оценки в глобална и прилага операция, която е хомоморфна на структурата на моделите.

Това позволява изграждане на абстрактни „агрегиращи семантики“.

2.5. Семиринги, автоматни редукции и weighted граматики (Hebert & Simmons, Kuich & Salomaa)

Семиринговата структура:

$$(S, \oplus, \otimes, 0, 1)$$

е фундаментална в теорията на тегловни автомати [20], граматики с тежести, минимакс алгебри [21] и оптимизационни алгоритми върху графи.

Редукционните процеси в семиринга са точно:

$$\bigoplus_{x \in X} f(x),$$

което е частен случай на $A \circ^* (X, c)$ с: $G(x, c) = True$, $\odot = \oplus$ и $B(X)$ – произволно дърво (асоциативност на $\oplus$).

2.6. Денотационни семантики и домейнна теория (Scott, Strachey, Winskel)

В домейнната теория на Скот [25], редукционни оператори от вида:

$$F(X) = \bigsqcup_{+} \{f(x) \mid x \in X\}$$

са стандартни при дефиниране на фиксирани точки, рекурсивни семантики и итеративни изчисления.

Агрегаторът $A\circ^*$ с подходящи избори за $\odot$ и $B(X)$ реализира точно тези случаи.

2.7. Обобщаваща позиция

Обзорът показва, че преобразуванията:

- от логика → комбиниране чрез $\wedge$,$\vee$;
- от категории → свободни моноиди и универсални свойства;
- от алгебра на програмирането → fold/катаморфизми;
- от институции → абстрактни семантики;
- от семиринги → общи редукции $\bigoplus$;
- от домейнната теория → фиксирани точки;
- от граматики → композиции на структурни единици.

използват оператори, които в различна степен съвпадат с агрегаторната схема.

3. Формален модел на агрегатора $A\circ$ и разширението $A\circ^*$

В този раздел се представя единната формална рамка на агрегатора, която включва базовия оператор $A\circ$, въведен в [28], и неговото строго съвместимо разширение $A\circ^*$, разгърнато в [29]. Моделът се основава върху ясно дефинирани параметри: входна структура, контекст, оценителна функция, предикат за избор, бинарна операция и редукционно дърво. Всички понятия се дефинират независимо от конкретна алгебрична, логическа или категорийна интерпретация.

3.1. Дефиниране на входния домейн

Нека D е непразно множество от дискретни елементи.

Дефинираме клас от крайни дискретни структури:

$$P_f^*(D)$$

където:

$P_f(D)$ – всички крайни подмножества на D;

$P_{flist}(D)$ – всички крайни списъци над D;

$P_{ftree}(D)$ – всички крайни дървовидни структури;

$P_{fgraph}(D)$ – всички крайни насочени или ненасочени графи;

$P_{fcomb}(D)$ – комбинации или вложени структури.

Следователно:

$P_f^*(D) = P_f(D) \cup P_{flist}(D) \cup P_{ftree}(D) \cup P_{fgraph}(D) \cup P_{fcomb}(D)$.

Входният аргумент на агрегатора е произволно $X \in P_f^*(D)$.

3.2. Контекстно множество

Нека C е множество от контексти. Контекстът $c \in C$ представлява параметри на средата, глобално състояние, външни зависимости, структурни ограничения или частично решение.

Така оценяването и изборът на активни елементи могат да зависят от контекстът c.

3.3. Оценителна функция

Оценителната функция е:

$$f: D \times C \to S,$$

където S е носещ домейн, например реални числа R, булеви стойности $\{True, False\}$, думи Σ^*, функции $\Sigma \to \Sigma$ или елементи на алгебрична структура.

Така за всяко $x \in X$ и $c \in C$ се получава оценка $f(x, c) \in S$.

3.4. Предикат за избор

Предикатът за избор (филтър) е:

$G: D \times C \to \{True, False\}$.

Активните елементи на X се определят чрез:

$Act(X, c) = \{x \in X \mid G(x, c) = True\}$.

Филтърът позволява условно участие на елементи, избор според контекст и динамична редукция на входа.

3.5. Носеща алгебра

Бинарната операция на агрегация е част от алгебрична структура:

$(S, \odot, e)$,

където: $\odot: S \times S \to S$ е тотална бинарна операция;

$e \in S$ е неутрален елемент, т.е.

$e \odot a = a$ и $a \odot e = a$.

Операцията може да бъде асоциативна / неасоциативна, комутативна / некомутативна или идемпотентна / неидемпотентна.

Този общ поглед е важен, защото агрегаторът допуска параметризация върху произволна такава алгебра.

3.6. Структуриращо правило $B(X)$

Критичен елемент в разширената дефиниция е:

$B: P_f^*(D) \to T$, където T е клас от крайни редукционни дървета.

Дървото $B(X)$ определя реда на комбиниране на елементите, групиране със скобите (структурата на редукцията), подразделянето на X на подструктури и възможни вложени редукции ако $B(X)$ е:

- линейна верига → получава се fold;
- пълно двоично дърво → паралелна редукция;
- дърво на синтактичен анализ → граматична редукция;
- DAG → графова редукция.

3.7. Базов агрегатор $A \circ$

При класическа конфигурация:

$C = \{\star\}$ (неутрален контекст),

$P_f^*(D) = P_f(D)$,

$\odot$ е асоциативна и комутативна.

Дефинираме: $A \circ [x \in X: G(x)] f(x) = f(x_1) \odot f(x_2) \odot \cdots \odot f(x_k)$,

където X е множество, а редът е без значение.

3.8. Разширен агрегатор $A \circ^*$

Разширението не въвежда нов оператор, а добавя нови параметри.

Дефиниция: За всеки $X \in P_f^*(D)$ и контекст $c \in C$:

$A \circ^* (X, c) = B(X)(f(x_1, c), f(x_2, c), \ldots, f(x_k, c))$,

където: $x_1, \ldots, x_k \in Act(X, c)$.

$B(X)(\cdot)$ означава - прилагай последователно операцията $\odot$ точно в зададения ред и структура.

Така $A \circ = A \circ^*$ в частния случай на комутативна асоциативна операция.

4. Сравнителен анализ между агрегатора $A\circ$, разширения модел $A\circ^*$ и класическите теоретични подходи

Целта на този раздел е да изведе формален, структурен и концептуален анализ на универсалния агрегатор в контекста на класическите теории: логика, институции, категорийни конструкции, алгебра на програмирането, семиринги и домейнни модели. Подходът е сравним и последователен: за всяка теория се извеждат (1) ключовите нейни оператори; (2) начинът, по който те представляват частни случаи на $A\circ^*$; (3) свойствата, които се запазват или обобщават в агрегаторната схема.

4.1. Сравнение с булева логика и логиката на Kleene

В класическата булева логика [1], [3]:

- истинностните стойности принадлежат към домейна $\{True, False\}$,
- основните бинарни операции са ∧ (конюнкция) и ∨ (дизюнкция),
- универсалният квантор ∀ и екзистенциалният ∃ могат да бъдат дефинирани чрез редукция.

Формално:

$$\forall x \in X\ P(x) \equiv \bigwedge_{x\in X} P(x), \exists x \in X\ P(x) \equiv \bigvee_{x\in X} P(x).$$

Тези два оператора са частни случаи на $A\circ^*$ при:

$D = X, C = \{\star\}$,

$f(x,c) = P(x)$,

$G(x,c) = True$,

$\odot$=∧ или $\odot$=∨,

$B(X)$ — произволна структура (поради асоциативност).

Следователно булевата логика е естествен подмодел на агрегаторната рамка.

4.2. Сравнение с институциите на Гогуен–Бърстал

Институциите [16] формализират логиката като четворка:

$$I = (Sign, Sen, Mod, \vDash)$$

където:

Sign — сигнатури,

Sen — синтактични изрази,

Mod — модели,

$\vDash$ — релация на удовлетворимост.

Агрегаторът $A\circ^*$ взаимодейства с институционалния модел, като комбинира множество оценки на формули или модели, допуска различни структури на комбиниране $B(X)$, позволява условно участие на елементи чрез $G(x,c)$ и свързва логическото оценяване с алгебрична редукция.

С други думи $A\circ^*$ (X,c) може да реализира семантика на удовлетворимост в обобщена форма.

Това разширява традиционния институционален подход, без да го нарушава.

4.3. Сравнение с категорийни конструкции (Mac Lane, Lawvere)

В категориите редукционните оператори се отразяват чрез моноидни обекти, свободни моноиди и универсални свойства.

Моноидът $(S,\odot,e)$ играе централна роля в дефиницията на агрегатора. В категорийна форма:

- $A\circ$ е хомоморфизъм от свободния комутативен моноид върху X в S;
- $A\circ^*$ е редукционен функтор, параметризиран от $B(X)$.

– Агрегаторът обединява комутативни редукции (като $\sum$, $\prod$), некомутативни композиции ($\odot$ като функция или морфизъм) и дървовидни компоновки чрез $B(X)$.

Категорийната теория обяснява универсалното свойство на агрегатора.

4.4. Сравнение с „Алгебра на програмирането“ (Bird & de Moor)

Катаморфизмите (fold) [11] са оператори от вида:

$$\text{fold}_{\odot}: D^* \to S.$$

Това е точно случаят, в който входът е списъчна структура, $\odot$ е асоциативна, редът е предварително зададен.

Агрегаторът $A \circ^*$ е по-общ, защото:

– допуска произволни структури $X \in P_f(D)$,

– допуска контекстна зависимост,

– допуска условен избор чрез G,

– допуска локални композиционни структури $B(X)$.

Следователно:

$$\text{fold} \subset A \circ \subset A \circ^*.$$

4.5. Сравнение със семирингови редукции

В семиринг $(S, \oplus, \otimes, 0, 1)$ най-често срещаната редукция е:

$$\bigoplus_{x \in X} f(x).$$

Това е класическата weighted редукция, използвана при автомати с тежести [20], графови алгоритми, минимакс алгебри [21] и др.

Този оператор е изцяло подмножество на $A \circ^*$ при:

$\odot = \oplus$,

$B(X)$ — произволно асоциативно дърво,

$G(x, c) \equiv True$.

4.6. Сравнение с домейнната теория и семантики на Скот–Стрейчей

Операторите за вземане на инфимум/супремум в домейните [25] са частен случай при $\odot = \sqcup$ или $\sqcap$.

Агрегаторът допуска тези операции, допуска частично подредени множества и допуска фиксирани точки чрез итерации на $A \circ^*$.

Обобщение

Разширеният агрегатор обединява логическите квантори. Включва институционални редукции, развива категорийни хомоморфизми. Той надхвърля класическите fold-ове и включва всички семирингови редукции. Освен това реализира домейнни оператори.

Това доказва универсалния характер на агрегатора.

5. Предимства, граници и разширяемост на агрегатора $A \circ$

В този раздел се изтъкват силните страни и границите на универсалния агрегатор $A \circ$, както и начините, по които разширеният модел $A \circ^*$ разширява функционалността му, без да нарушава първоначалната му дефиниция. Всички изводи се основават на формалните параметри, въведени в Раздел 3, и на сравнителния анализ от Раздел 4.

5.1. Предимства на агрегатора

5.1.1. Универсалност върху дискретни структури

Агрегаторът $A\circ$ реализира универсален модел за редукция върху крайни множества. Благодарение на:

$$X \in P_f^*(D),$$

и на това, че редукцията използва обща операция $\odot$, агрегаторът е способен да обедини множество класически оператори ($\sum$, $\prod$, $\wedge$, $\vee$, min, max), да формализира структурни комбинации и да се адаптира към различни типове вход (множества, списъци, дървета, графи).

Той представлява естествено обобщение на класическите fold-ове и семирингговите редукции.

5.1.2. Еднородност и параметрична гъвкавост

Операторът е изцяло параметризиран чрез:

- избор на активни елементи G,
- оценка f,
- структура B,
- операция $\odot$,
- контекст c.

Тази параметрична структура позволява включване на логически квантори ([1], [3]), включване на семирингови операции ([19]), реализация на категорийни хомоморфизми ([5]) и моделиране на институционални семантики ([16]).

5.1.3. Формална стабилност

Агрегаторът притежава ясно дефинируеми свойства:

- асоциативност (когато $\odot$ е асоциативна),
- комутативност (когато $\odot$ е комутативна),
- инвариантност на структурата (когато $B(X)$ е стабилно определено),
- монотонност (когато $\odot$ е монотонна операция в подреден домейн).

Така агрегаторът може да бъде използван в домейнни модели ([25], [26]).

5.2. Граници на базовия модел и тяхното разширяване в $A\circ^*$

5.2.1. Използване на агрегатора при рекурсивни процеси

Базовият агрегатор $A\circ$ не дефинира самостоятелно рекурсивност, тъй като работи върху крайни структури. Разширената дефиниция позволява:

$$A\circ^*(X,c) = B(X)(f(x,c)),$$

и може да бъде използвана рекурсивно чрез вложена агрегация, итеративно прилагане до фиксирана точка, рекурсивно дефинирани структури (дървета, графи) и други рекурсивни процеси.

Този подход е съвместим с домейнната теория на Скот ([25]).

5.2.2. Функциониране върху неасоциативни операции

Когато $\odot$ не е асоциативна, класическият модел не е дефиниран еднозначно. В разширената схема еднозначността се гарантира чрез $B(X)$, което фиксира конкретно подреждане и групиране със скоби. Така неасоциативни операции са разрешени, операторът остава детерминиран, а структурата на редукцията става част от дефиницията.

5.2.3. Управление на зависими стойности

Когато е необходимо състоянието да се акумулира или да се изменя по време на редукцията, се използва разширената оценителна функция:

$$f: D \times C \to S,$$

където контекстът C може да бъде глобален, локален или актуализиран между стъпките.

Така агрегаторът работи като обобщен fold със състояние.

5.2.4. Интегриране на логически операции

Класическите оператори за импликация, отрицание и еквивалентност се реализират чрез разширяване на домейна S, избор на подходяща операция $\odot$ и дефиниране на подходяща функция $f(x, c)$.

Например:

$$P(x) \Rightarrow Q(x) = \neg P(x) \vee Q(x)$$

е частен случай при $\odot = \vee$.

5.2.5. Управление на структурата на комбиниране

Най-фундаменталната граница на базовия модел (липса на контрол над реда) е напълно елиминирана в разширената версия чрез:

$B(X): P_f^*(D) \to T$,

където T е клас от дървета.

Това позволява различни стратегии на редукция, оптимизационни цели, паралелни редукции, граматични структури и графови редукции.

Така $A \circ^*$ става универсален оператор за композиция.

5.3. Синтез: разширената схема като обединяващ модел

Разширеният агрегатор запазва оригиналния оператор $A \circ$, Той добавя нови параметри (G, B, контекст), запазва всички класически свойства и обединява (като специални случаи) логика, категории, семиринги, fold-ове и домейнни оператори.

Това го прави универсален редукционен оператор за дискретни структури.

6. Потенциални приложения на агрегатора $A^{\circ *}$

Разширената дефиниция на агрегатора $A^{\circ *}$, формулирана в Раздел 3, предоставя общ алгебричен и логически апарат за работа с дискретни структури. Поради наличието на филтър $G(x, c)$, оценителна функция $f(x, c)$, бинарна операция $\odot$, редукционно дърво $B(X)$ и контекст c операторът може да се прилага в широк спектър от области. Тук представяме ключовите направления, в които агрегаторът действа като естествена обединителна рамка.

6.1. Логически системи и обобщени квантори

Агрегаторът позволява дефиниране на:

- класическите квантори ∀ и ∃,
- обобщени квантори (Most, Exactly k, At least k),
- логически операции върху множества от формули,
- агрегиращи семантики на институционални логики ([16]).

Пример: обобщен квантор „повечето“ може да бъде реализиран чрез:

$$A^{\circ *}(X, c) = \begin{cases} True & \text{ако } |\{x \in X \mid P(x)\}| > \dfrac{|X|}{2}, \\ False & \text{при останалите случаи.} \end{cases}$$

Така $A^{\circ *}$ служи като универсален модел за оператори в логиката.

6.2. Семантика на програмни езици и редукционни модели

Семантиките на Скот–Стрейчей ([25], [26]) често се опират на операции по вземане на супремум, редукции в частично подредени множества или фиксирани точки на функции.

Агрегаторът $A^{\circ *}$ реализира тези операции като:

$$\biguplus_{x \in X} f(x,c) \leftrightarrow A^{\circ *}(X,c) \text{ при } \odot = \sqcup.$$

По този начин $A^{\circ *}$ може да изразява семантики на ламбда-изчислението, семантики на езиците от високо ниво и редукционни процеси в функционални езици.

6.3. Графови алгоритми и оптимизационни конструкции

Голям клас графови задачи могат да се формулират чрез агрегатор – най-къс път (min-редукция), преброяване на пътища (sum-редукция), максимална стойност по ребрата (max-редукция), Bellman–Ford, Dijkstra, Floyd–Warshall както и много други.

Пример:

$$A^{\circ *}(Paths(v \to u), c)$$

където:

- $Paths(v \to u)$ е множество от всички пътища от v до u,
- $f(p,c)$ е теглото на пътя p,
- $\odot = \min$,

B определя дали редукцията е паралелна или итеративна.

Така агрегаторът се превръща в универсален механизъм за графови оптимизации.

6.4. Семирингови изчисления и weighted автомати

В семиринг $(S, \oplus, \otimes, 0, 1)$:

$$A^{\circ *}(X,c) = \bigoplus_{x \in X} f(x,c)$$

е стандартната редукция за weighted автомати ([20]). Това позволява реализация на алгоритми за разпознаване на weighted езици, оценяване на вероятностни системи, минимакс алгебри ([21]) и динамично програмиране върху формални структури.

Агрегаторът действа като общо ядро при такива редукции.

6.5. Категорийни конструкции и естествени трансформации

В категорийния контекст агрегаторът реализира хомоморфизъм от свободен моноид в носещата алгебра. При некомутативни случаи той определя конкретна композиционна структура чрез $B(X)$. Обобщава функтори, редуциращи списъци, дървета или диаграми.

Така той може да бъде използван за създаване на нови моноидни обекти, дефиниране на трансформации между агрегиращи функции или други универсални факторизации на композиции.

6.6. Граматики, синтактични анализи и композиции

В граматиките от вида:

$$G = (V, \Sigma, P, S)$$

агрегиращите операции играят роля във: комбиниране на вероятности в вероятностни граматики, комбиниране на тегла в weighted CFG и при синтактични композиции в синтактични дървета.

Чрез подходящо $B(X)$ агрегаторът може да служи като обобщение на синтактичен анализ, оператор за оценяване на структура и оператор за комбиниране на алтернативи.

6.7. Функционално програмиране и модели на изчисления

Във функционалните езици fold, reduce, map-reduce са стандартни оператори. Всички могат да се изразят чрез:

$$A^{\circ *}(X,c).$$

Примери:

- sum = агрегатор със $\odot = +$,
- product = агрегатор със $\odot = \cdot$,
- all = агрегатор със $\odot = \wedge$,
- any = агрегатор със $\odot = \vee$,
- concatenate = агрегатор със $\odot =$ concat.

Разширеният агрегатор добавя вложени редукции, контекст, неасоциативни операции, динамично определени структури на композиция.

Това е по-мощно от всеки стандартен fold.

6.8. Обобщаваща роля в математическите и изчислителни науки

Агрегаторът $A^{\circ *}$:

- обединява логика, алгебра, категории, семантики, автоматни модели и графови алгоритми;
- действа като единен оператор за редукция в дискретни системи;
- предоставя формална основа за изграждане на нови езици, семантики и алгоритми.

7. Математически анализ на агрегатора $A\circ$ и разширената схема $A\circ^*$

Този раздел представя математически анализ на агрегатора, като целта е да се формализират: (1) структурните свойства на оператора, (2) влиянието на параметрите $G, f, \odot, B, c$, (3) ролята на домейните $P_f^*(D)$ и носещата алгебра $(S,\odot, e)$.

Важно: агрегаторът не се променя; разширената версия $A\circ^*$ е надстройка, която включва базовия модел като частен случай.

7.1. Формална структура на оператора

7.1.1. Основни компоненти

Агрегаторът е дефиниран чрез пет основни параметъра:

- Входна структура – $X \in P_f^*(D)$
- Филтърна функция – $G: D \times C \to \{True, False\}$
- Оценителна функция – $f: D \times C \to S$
- Бинарна операция – $\odot: S \times S \to S$
- Редукционно дърво – $B: P_f^*(D) \to T$

7.2. Основна дефиниция

7.2.1. Базов агрегатор $A\circ$

При комутативна и асоциативна операция $\odot$, множество X, и филтър G

$A\circ[X] = \odot_{x\in X, G(x)=True} f(x)$

където $\odot$ означава последователно прилагане на $\odot$.

7.2.2. Разширена дефиниция $A\circ^*$

Разширението въвежда редукционна структура $B(X)$ и контекст с:

$$A\circ^*(X,c) = B(X)(f(x_1,c), f(x_2,c), \dots, f(x_k,c))$$

където:

$\{x_1, \dots, x_k\} = Act(X,c) = \{x \in X \mid G(x,c) = True\}$.

Тук $B(X)(\cdot)$ указва недвусмислено реда и структурата на комбиниране.

7.3. Еднозначност и тоталност

Теорема 7.1 Еднозначност.

Ако $B(X)$ е крайно дърво, $\odot$ е тотална операция и f е тотална функция,

тогава $A \circ^* (X, c)$ е еднозначно определен.

Доказателство: Всеки лист е стойност $f(x_i, c)$. Всеки вътрешен възел прилага $\odot$ върху две вече дефинирани стойности. Крайно множество от крайни операции гарантира уникална дефиниция.

7.4. Колапс към комутативно-асоциативната форма

Ако $\odot$ е: асоциативна и комутативна, а $B(X)$ е произволно дърво над X, тогава:

$$A \circ^* (X, c) = \odot_{x \in Act(X,c)} f(x, c).$$

Това означава, че разширението не променя резултата, ако редът е без значение.

7.5. Роля на предиката за избор G

Филтърът $G(x, c)$ определя домейна на редукцията:

$$Act(X, c) = \{x \in X \mid G(x, c) = True\}.$$

Поради тази причина агрегаторът представя частични редукции, условни редукции и контекстно-управлявани редукции.

7.6. Роля на оценителната функция f

Оценката определя трансформацията:

$$x \mapsto f(x, c).$$

f може да бъде скаларна, булева, функционална, структурна, операторна.

Това позволява агрегиране на числа, формули, функции, граматики, маршрути и графови пътища.

7.7. Бинарна операция $\odot$

Тя задава начина на комбиниране:

– сума → $\odot = +$
– произведение → $\odot = \cdot$
– минимум → $\odot = \min$
– максимум → $\odot = \max$
– конюнкция/дизюнкция → $\odot = \wedge, \vee$
– конкатенация → $\odot =$ concat
– композиция → $\odot = \circ$

Изборът ѝ определя класа на агрегацията.

7.8. Редукционно дърво $\boldsymbol{B(X)}$

$B(X)$ определя структурата - линейно → fold, двоично → паралелна редукция, дълбоко дърво → граматика, DAG → графова редукция, един възел → неутрален случай (единична стойност) или друга подобна сруктура.

Теорема 7.2 - При фиксирана $B(X)$, дори неасоциативни операции дават детерминиран резултат.

7.9. Инвариантност

Ако $\odot$ е комутативна и асоциативна:

$$B(X_1) = B(X_2) \Rightarrow A \circ^* (X_1, c) = A \circ^* (X_2, c)$$

ако X_1 и X_2 са пермутации на едни и същи елементи.

7.10. Монотонност

Ако S е частично подредено множество и $\odot$ е монотонна функция:

$a \leq b \Rightarrow a \odot c \leq b \odot c$,

и аналогично за втория аргумент, тогава:

$$X \subseteq Y \Rightarrow A \circ^* (X, c) \leq A \circ^* (Y, c)$$

при равни G, f и B.

8. Теореми и доказателства за агрегатора $A \circ^*$

Тук ще разгледаме основните математически теореми, които характеризират поведението на агрегатора $A \circ^*$. Те доказват неговата устойчивост, универсалност и структурна съвместимост с широк спектър от алгебрични, логически и категорийни конструкции. Доказателствата са формулирани за разширения модел, като класическият агрегатор $A \circ$ се получава като специален случай.

8.1. Теорема за еднозначността Теорема 8.1 (Еднозначност на резултата).

Нека:

$X \in P_f^*(D)$ е крайна структура,

$c \in C$ е произволен контекст,

$f: D \times C \to S$ е тотална функция,

$G: D \times C \to \{True, False\}$ е тотален филтър,

$\bigodot: S \times S \to S$ е тотална бинарна операция,

$B(X)$ е крайно редукционно дърво.

Тогава агрегаторът $A \circ^* (X, c)$ е еднозначно определена стойност.

Доказателство.

Множество от активни елементи е крайно:

$Act(X, c) = \{x \in X \mid G(x, c) = True\}$ е крайно, тъй като X е крайно.

Всяко листо в дървото $B(X)$ получава стойност $f(x, c)$, която е определена, тъй като f е тотална.

Всеки вътрешен възел прилага бинарната операция $\bigodot$, която е дефинирана за всички двойки аргументи.

Тъй като дървото е крайно и всяка операция е тотална, резултатът в корена е една стойност от S.

8.2. Теорема за редукция до комутативно-асоциативен случай

Теорема 8.2. Ако $\bigodot$ е асоциативна и комутативна, тогава:

$A \circ^* (X, c) = \bigodot_{x \in Act(X,c)} f(x, c)$,

където $\bigodot$ обозначава произволно последователно прилагане на $\bigodot$ върху всички активни оценки.

Доказателство.

При асоциативност групиране със скобите е без значение. При комутативност редът е без значение. Следователно дървото $B(X)$ не влияе върху резултата.

8.3. Теорема за стабилност под пермутации

Теорема 8.3.

Ако $\bigodot$ е комутативна и асоциативна и X, Y са пермутации на едни и същи елементи, тогава:

$A \circ^* (X, c) = A \circ^* (Y, c)$.

Доказателство.

Произтича пряко от Теорема 8.2.

8.4. Теорема за монотонност

Теорема 8.4.

Нека:

$(S, \leq)$ е частично подредено множество,

$\bigodot$ е монотонна операция, т.е.

$a \leq b \Rightarrow a \bigodot c \leq b \bigodot c, c \bigodot a \leq c \bigodot b$,

G, f, B са фиксирани.

Тогава:

$X \subseteq Y \implies A \circ^* (X, c) \leq A \circ^* (Y, c)$.

Доказателство.

Увеличаването на входното множество може само да добави повече стойности $f(x, c)$ към редукцията, а монотонността гарантира, че това не намалява резултата.

8.5. Теорема за разширимост чрез контекст

Теорема 8.5.

Функция от вида:

$$X \mapsto A \circ^* (X, c)$$

е параметризируема чрез контекст c и може да представлява семейство от хомоморфизми:

$$A \circ^* (-, c_1), A \circ^* (-, c_2), \ldots$$

които се различават само по локалните оценки.

Доказателство.

Контекстът участва единствено чрез $G(x, c)$ и $f(x, c)$. Това променя стойностите в листата, без да променя структурата $B(X)$, което означава, че агрегацията запазва алгебричната си форма, но с различни параметри. ■

8.6. Теорема за редукции върху дървовидни структури

Теорема 8.6.

Ако $B(X)$ е дърво, получено чрез декомпозиция на X в подструктури $X_1, \ldots, X_n$, тогава:

$A \circ^* (X, c) = A \circ^* (X_1, c) \odot A \circ^* (X_2, c) \odot \cdots \odot A \circ^* (X_n, c)$.

Доказателство.

Дървото $B(X)$ реализира точно тази факторизация, като всеки възел представлява композиране на стойностите на поддърветата.

8.7. Теорема за вложена агрегация

Теорема 8.7.

При вложена структура:

$$X = \{X_1, \ldots, X_m\}$$

и съвместима B, имаме:

$A \circ^* (X, c) = B(X)(A \circ^* (X_1, c), \ldots, A \circ^* (X_m, c))$.

Тази формула обобщава катаморфизмите и показва рекурсивната природа на агрегатора.

8.8. Универсална теорема за представимост

Теорема 8.8 (Универсалност).

Всеки краен редукционен оператор R, който избира подмножество от елементи, преобразува ги чрез функция, и ги комбинира чрез бинарна операция може да бъде представен като агрегатор $A \circ^*$.

Доказателство.

Конструираме:

- G като предикат за избора в R,
- f като трансформацията в R,
- $\odot$ като комбиниращa операция в R,
- $B(X)$ като структурата на редукцията в R.

Тогава R и $A \circ^*$ съвпадат по дефиниция.

Разширени свойства, примери и специални случаи на агрегатора $A\circ^*$

Тук ще разгледаме разширени структурни свойства на агрегатора $A\circ^*$, както и конкретни примери, които демонстрират неговата универсалност и способността му да обединява класически оператори от логика, алгебра, програмиране, семантики и оптимизационни алгоритми.

Целта е да се покаже, че $A\circ^*$ не е просто „общ оператор“, а унифициращ формализъм с ясно дефинирана математическа структура.

9.1. Обобщение на класическите редукции чрез един оператор

Много добре познатите редукции сума, произведение, минимум, максимум, булеви свързки $\wedge$ и $\vee$, конкатенация на думи, композиция на функции, редукции в семиринги $\oplus,\otimes$, инфимум/супремум в домейнни модели се формализират чрез една и съща схема:

$$A\circ^*(X,c) = B(X)(f(x_1,c),\dots,f(x_k,c)),$$

където изборът на $\circ$ определя конкретния оператор.

9.2. Свойство: неутралност и единични структури

Ако e е неутралният елемент на $\circ$, тогава:

$$A\circ^*(\emptyset,c) = e.$$

За единично множество:

$$A\circ^*(\{x\},c) = f(x,c).$$

Това показва, че агрегаторът притежава правилното поведение за празни входове и минимални входни структури.

9.3. Свойство: разпределимост върху декомпозиции

Нека X се разложи в подструктури:

$$X = X_1 \cup X_2 \cup \dots \cup X_n,$$

и $B(X)$ е съвместима с тази декомпозиция. Тогава:

$$A\circ^*(X,c) = A\circ^*(X_1,c)\circ A\circ^*(X_2,c)\circ\dots\circ A\circ^*(X_n,c).$$

Това е фундаментално за divide-and-conquer алгоритми, паралелни редукции, синтактични анализи и факторизирани структури.

9.4. Свойство: рекурсивна дефинируемост

Благодарение на рекурсивната природа на $B(X)$, агрегаторът може да бъде дефиниран рекурсивно:

$$A\circ^*(X,c) = \begin{cases} e, & X=\emptyset,\\ f(x,c), & X=\{x\},\\ A\circ^*(X_L,c)\circ A\circ^*(X_R,c), & X=\text{композиция на } X_L,X_R. \end{cases}$$

Този модел обединява катаморфизми (fold), синтактични дървета, рекурсивни дефиниции в функционални езици, операции върху формални граматики.

9.5. Пример 1: Булеви квантори

Нека:

$$S = \{True, False\},$$

$\circ=\wedge$ или $\vee$,

$f(x,c) = P(x)$.

Тогава:

$$A\circ^*(X,c) = True \iff \forall x\in X\ P(x)$$

и

$$A\circ^*(X,c) = True \iff \exists x\in X\ P(x).$$

Това показва, че агрегаторът включва класическата логика като частен случай.

9.6. Пример 2: Семиринговi редукции

Нека:

$$S = (R, \min, +, \infty, 0) \text{ – тропичен семиринг,}$$

$f(x, c)$ е дължината на ребро.

Тогава:

$$A \circ^* (Paths(v \to u), c) = \min_{p \in Paths(v \to u)} \sum_{\text{edges } e \in p} w(e).$$

Това е точно графовата редукция за най-къс път (Dijkstra/Bellman–Ford).

9.7. Пример 3: Функционални композиции

Нека:

$$S = Func(A, A),$$

$\circ{=}\circ$ (композиция).

Тогава:

$$A \circ^* (X, c) = f(x_1, c) \circ f(x_2, c) \circ \cdots \circ f(x_k, c).$$

Това формализира последователни трансформации, pipeline модели, изчислителни композиции.

9.8. Пример 4: Конкатенация на думи

Нека:

$$S = \Sigma^*,$$

$\circ$ = конкатенация.

Тогава агрегаторът е:

$$A \circ^* (X, c) = w_1 w_2 \ldots w_k.$$

Централно в граматики, синтактични анализи и езикови модели.

9.9. Пример 5: Логико-алгебрични зависимости

Ако:

$G(x, c) = P(x, c)$ е предикат,

$f(x, c) = Q(x, c)$ е друга оценка,

$\circ{=}\Rightarrow$ (импликация),

тогава агрегаторът представлява глобална логическа импликация върху множество формули.

9.10. Пример 6: Оценяване на вероятностни модели

Нека:

$$S = [0,1],$$

$\circ{=}\cdot$,

$f(x, c) = P(x)$.

Тогава:

$$A \circ^* (X, c) = \prod_{x \in X} P(x)$$

е вероятността независими събития да настъпят едновременно.

9.11. Специален случай: Агрегаторът като обобщение на fold

При линейна структура:

$$B(X) = (x_1 \circ (x_2 \circ (\ldots))).$$

Получава се точно операторът fold от алгебрата на програмирането [11].

9.12. Специален случай: некомутативни и неасоциативни операции

Агрегаторът позволява некомутативни операции (редът се задава чрез B) и неасоциативни операции (групиране със скобито се задава чрез B). По този начин той надхвърля класическите редукции.

Заключение

Разработената в настоящата монография теория на универсалния агрегатор $A\circ$ и неговото разширение $A\circ^*$ демонстрира, че широк спектър от класически оператори в логиката, алгебрата, категорийния анализ, семантиката на програмни езици, оптимизационните алгоритми и теорията на формалните езици могат да бъдат обединени чрез единна, строго дефинирана рамка.

Основните резултати могат да се обобщят така:

10.1. Универсалност на модела

Разширената дефиниция:

$$A\circ^*(X,c) = B(X)(f(x_1,c),\dots,f(x_k,c)),$$

показва, че агрегаторът:

- може да работи върху произволни дискретни структури $Pf^*(D)$;
- допуска контекстно зависими оценки;
- реализира условен избор чрез предиката G;
- поддържа неасоциативни и некомутативни операции чрез $B(X)$;
- обобщава всички класически комутативно-асоциативни редукции.

Това го прави универсален оператор за дискретни редукции.

10.2. Обединяване на класическите теории

Агрегаторът включва като специални случаи:

- булеви квантори на Kleene и Tarski ([1], [3]),
- абстрактни логики и институции ([16]),
- моноидни и категорийни конструкции на Mac Lane ([5]),
- катаморфизми и fold оператори ([11]),
- семирингови редукции ([19], [20]),
- операции в домейнната теория (Scott–Strachey) ([25]).

Всяка от тези теории се описва чрез стандартна схема:

$$x \mapsto f(x,c),\ \text{select}(x) = G(x,c),\ \text{combine using}\ \odot,$$

което е точно структурата на агрегатора $A\circ^*$.

10.3. Еднозначност, стабилност и математически свойства

В раздели 7 и 8 бяха доказани :

- еднозначност на дефиницията (Теорема 8.1),
- стабилност под пермутации за комутативни операции (Теорема 8.3),
- монотонност (Теорема 8.4),
- структурирана редукция чрез дървета (Теорема 8.6),
- възможност за вложена агрегация (Теорема 8.7),
- универсална представимост (Теорема 8.8):

Всеки краен редукционен оператор може да бъде формално представен като $A\circ^*$.

Това установява ролята на агрегатора като фундаментален механизъм за обединяване на редукционните структури.

10.4. Практическа приложимост

На база на направените по-горе изводи, може да се каже, че агрегаторът намира приложение в логически системи и квантори, семантики на програмни езици, graph-based оптимизационни алгоритми, weighted автомати и граматики, функционални езици и map-reduce схеми, анализ на структури, формални доказателства и моделиране.

Той представлява естествена основа за формални езици, оптимизационни системи и програмни парадигми, изградени около едно ядро: дискретна редукция чрез множество механизми.

10.5. Концептуално значение

Представената теория показва, че не е необходим набор от отделни оператори за сума, произведение, екстремуми, квантори и др. Един-единствен оператор $A\circ^*$ може да обедини всички тях. Различните агрегиращи процеси са просто параметризации на една обща конструкция.

Това представлява нов обединяващ поглед към дискретните структури в математиката и информатиката.

Речник на използваните в монографията означения

1. Основни променливи и множества

D – Домейн; множество от всички елементи x, върху които агрегаторът може да действа.

X – Конкретна входна структура (множество, списък, дърво, граф).

C – Множество от контексти.

c – Конкретен контекстен параметър.

S – Носещ домейн на стойностите (целевото множество, в което се комбинират оценки).

T – Множество/клас от редукционни дървета, използвани от $B(X)$.

$\mathrm{Pf}(D)$ – Множество от всички крайни подмножества на D.

$\mathrm{Pf}^*(D)$ – Универсален набор от дискретни структури над D: множества, списъци, дървета, графи, вложени структури.

$\mathrm{Pf}_{\mathrm{list}}(D)$ – Всички крайни списъци над D.

$\mathrm{Pf}_{\mathrm{tree}}(D)$ – Всички крайни дървета над D.

$\mathrm{Pf}_{\mathrm{graph}}(D)$ – Всички крайни графи над D.

$\mathrm{Pf}_{\mathrm{comb}}(D)$ – Всички вложени/комбинирани структури над D.

2. Функции, оператори и предикати

$G(x,c)$ – Филтър (предикат). Определя дали елемент x участва в агрегацията при контекст c.

$\mathrm{Act}(X,c)$ – Множество от активни елементи след филтриране:
$A_c(X,c)=\{x\in X \mid G(x,c)=\mathrm{True}\}$.

$\mathrm{Act}(X,c)=\{x\in X \mid G(x,c)=\mathrm{True}\}$.

$f(x,c)$ – Оценителна функция. Превежда всеки елемент x в стойност от S, зависеща от контекст c.

$B(X)$ – Редукционно дърво (структуриращо правило). Определя точния ред и групиране на операциите $\odot$.

$A\circ$ – Базов агрегатор. Комбинира стойности $f(x)$, ако $\odot$ е комутативна и асоциативна.

$A \circ^* (X, c)$ – Разширен агрегатор.

$$A \circ^* (X, c) = B(X)(f(x_1, c), \dots, f(x_k, c)).$$

$A \circ^* (X, c) = B(X)(f(x_1, c), \dots, f(x_k, c))$.

$\odot$ – Операция на комбиниране върху S (например +, ×, min, max, ∧, ∨, конкатенация, композиция, $\oplus$).

e – Неутрален елемент на операцията $\odot$.

fold – Катаморфичен оператор (списъчна редукция). Частен случай на $A \circ$.

compose / $\circ$ – Композиция на функции (в контекста на оператори, не агрегатора).

3. Алгебрични структури

$(S, \odot, e)$ – Моноид (асоциативна операция + неутрален елемент).

Комутативен моноид – Моноид, при който $\odot$ е комутативна.

Моноиден обект (M, μ, η) – Обект в категория с операции:

μ – умножение

η – единичен морфизъм

$\mu: M \otimes M \to M$ – Умножение (категорийно).

$\eta: I \to M$ – Единичен морфизъм (категорийно).

4. Категорийни символи

$\otimes$ – Тензорно произведение (в категория).

I – Единичен обект (unit object).

$f: A \to B$ – Морфизъм.

$F: C \to D$ – Функтор.

$\mathrm{Nat}(F, G)$ – Множество от естествени преобразувания между функтори F и G.

$\simeq$ – Изоморфизъм на обекти/структури.

5. Символи от семиринги, домейнна теория и логика

Семиринги

$(S, \oplus, \otimes, 0, 1)$ – Семиринг:

$\oplus$ – адитивна операция

$\otimes$ – мултипликативна операция

0 – нула

1 – единица

$\bigoplus$ – Семирингова редукция:

$$\bigoplus_{x \in X} f(x).$$

$\bigoplus_{x \in X} f(x)$.

Домейнна теория

$\sqcup$ – Супремум (lub — least upper bound).

$\sqcap$ – Инфимум (glb — greatest lower bound).

$\mathrm{Fix}(F)$ – Фиксирана точка на F.

Логика и институции

$\models$ – Удовлетворимост:

Модел M удовлетворява формула φ.

Sen – Множество на синтактични изрази.

Mod – Множество на модели.

Sign – Множество на сигнатури.

6. Символи за класически агрегиращи операции

$\sum$ $f(x)$ – Сума. Частен случай на $A \circ$ при $\odot = +, e = 0$.

$\prod$ $f(x)$ – Произведение. Частен случай при $\odot = \times, e = 1$.

$\min f(x)$ – Минимум.

$\max f(x)$ – Максимум.

$\bigwedge$ $f(x)$ – Булево И. Частен случай с $\odot = \wedge$.

$\bigvee$ $f(x)$ – Булево ИЛИ.

++ – Конкатенация (свободен моноид).

$\circ$ – Композиция на функции.

7. Символи за структури в графовите приложения

$\mathrm{Paths}(v \to u)$ – Множество от всички пътища от връх v до u.

$w(e)$ – Тегло на ребро e.

$\sum_{e \in p} w(e)$ – Тегло на път p.

$\min_{p \in \mathrm{Paths}} (\dots)$ – Най-къс път.

Това е пример за реализация на Dijkstra/Bellman–Ford чрез $A \circ^*$.

8. Символи от теоремите за свойствата

$=$ – Еднозначно равенство.

$\leq$ – Частична наредба в $(S, \leq)$.

$\subseteq$ – Подмножество.

$B(X_1) = B(X_2)$ – Равенство на редукционни дървета.

$X \equiv Y$ – Пермутация на едни и същи елементи.

$\Rightarrow$ – Импликация.

$\Leftrightarrow$ – Еквивалентност.

9. Специализирани помощни символи

$\star$ – Неутрален / формален контекст (класически случай без контекст).

… – Продължение на редукцията.

$\vdash$ – Доказуемост .

$\varphi(x), \psi(x)$ – Логически формули.

$0, 1, \infty, -\infty$ – Константи в редукциите (за min/max).

ε – Празен низ (в свободния моноид над Σ).

Σ^* – Множество от всички думи над азбука Σ.

10. Пълен списък на агрегаторни специални случаи, както ги разпознава монографията

Оператор	Символ	Параметри	Забележки
Сума	Σ	$\odot = +, e = 0$	Комутативен моноид
Произведение	Π	$\odot = \times, e = 1$	Комутативен моноид
Мин	min	$\odot = \min, e = +\infty$	Идемпотентен
Макс	max	$\odot = \max, e = -\infty$	Идемпотентен
Булево И	$\bigwedge$	$\odot = \wedge, e =$ True	
Булево ИЛИ	$\bigvee$	$\odot = \vee, e =$ False	
Конкатенация		$++ \; e = \varepsilon$	Некомутативна

Композиция $\circ$ $e = \mathrm{id}$ Некомутативна, неасоциативна (в някои)
Семирингова редукция $\bigoplus$ $\odot = \bigoplus$ Общ модел
Тропическа редукция $\min/+$ $S = \mathbb{R} \cup \{\infty\}$ Графи, оптимизация

11. Символи за дървовидни / структурни редукции

$B(X)$ – Дърво, определящо структурата.
X_ℓ, X_r – Ляво и дясно поддърво (в теоремата за декомпозиция).
$(a \odot b)$ – Комбиниране на два листа/субагрегата.
$\mathrm{root}(B(X))$ – Корен на дървото.
$\mathrm{leaf}(x)$ – Лист, съдържащ стойност $f(x, c)$.

12. Употребявани логически символи

$\forall$ – Универсален квантор, реализуем чрез $\bigwedge$.
$\exists$ – Екзистенциален квантор, реализуем чрез $\bigvee$.
$\neg$ – Отрицание.
$\Rightarrow$ – Импликация.
$\mapsto$ – Отображение/проекция.

13. Мета-символи, използвани в описанията

"$\in$" – Принадлежност.
":" – Ограничение в агрегаторен контекст $(x \in D : G(x))$.
"$|\cdot|$" – Кардиналност.
"$\{\}$"/"" – Множество / списък.
σ – Пермутация (на X).
ℓ – Изброяване (списъчна форма) на множество.

Пълен съкратен речник

D – Домейн на елементите.
X – Входна дискретна структура (множество, списък, дърво, граф).
C – Множество от контексти.
c – Конкретен контекст.
$G(x, c)$ – Филтър (предикат).
$Act(X, c)$ – Множество от активни елементи.
$f(x, c)$ – Оценителна функция.
S – Целеви домейн.
$\odot$ – Бинарна операция в S.
e – Неутрален елемент на $\odot$.
$(S, \odot, e)$ – Моноид.
$B(X)$ – Редукционно дърво (структура на агрегиране).
$A\circ$ – Базов агрегатор.
$A\circ^*(X, c)$ – Разширен агрегатор.
$Pf(D)$ – Крайни множества над D.
$Pf^*(D)$ – Всички крайни структури над D.
μ – Моноидно умножение (категорийно).
η – Единичен морфизъм.
$\bigotimes$ – Тензорно произведение (категория/семиринг).
$\bigoplus$ – Адитивна операция в семиринг.

$\bigoplus$ – Семирингова редукция.
$\sqcap, \sqcup$ – Инфимум / супремум (домейнна теория).
$\vDash$ – Удовлетворимост (логика/институции).
φ, ψ – Формули.
Σ^* – Свободен моноид (думи).
ε – Празен низ.
$++$ – Конкатенация.
$\circ$ – Композиция на функции.
$Paths(v \to u)$ – Множество пътища (графи).
$w(e)$ – Тегло на ребро.
$0, 1, \infty, -\infty$ – Константи при редукции.
$\leq$ – Частична наредба.
$\subseteq$ – Подмножество.
$\simeq$ – Изоморфизъм.
σ – Пермутация.
ℓ – Изброяване.
$\bigwedge$, $\bigvee$ – Булеви агрегатори.
$\forall, \exists$ – Квантори.
$\neg$ – Отрицание.
$\Rightarrow, \Leftrightarrow$ – Импликация / еквивалентност.

References:

[1] KLEENE, S. C. Introduction to Metamathematics. North-Holland, 1952.
[2] KLEENE, S. C. Mathematical Logic. Wiley, 1967.
[3] TARSKI, A. Logic, Semantics, Metamathematics. Clarendon Press, 1956.
[4] CHURCH, A. A formulation of the simple theory of types. JSL, 1940.
[5] MAC LANE, S. Categories for the Working Mathematician. Springer, 1998.
[6] MAC LANE, S.; BIRKHOFF, G. Algebra. AMS Chelsea, 1988.
[7] BARR, M.; WELLS, C. Category Theory for Computing Science. 1999.
[8] EILENBERG, S.; MAC LANE, S. General theory of natural equivalences. TAMS, 1945.
[9] LAWVERE, F.; SCHANUEL, S. Conceptual Mathematics. Cambridge, 1997.
[10] LAWVERE, F. Functorial semantics of algebraic theories. PNAS, 1963.
[11] BIRD, R.; DE MOOR, O. The Algebra of Programming. Prentice Hall, 1997.
[12] MEIJER, E.; FOKKINGA, M.; PATERSON, R. FP with bananas, lenses and envelopes. 1991.
[13] MOGGI, E. Notions of computation and monads. I&C, 1991.
[14] WINSKEL, G. The Formal Semantics of Programming Languages. MIT Press, 1993.
[15] WINSKEL, G.; NIELSEN, M. Models for concurrency. 1995.
[16] GOGUEN, J.; BURSTALL, R. Institutions. JACM, 1984.
[17] GOGUEN, J. Sheaf semantics for interacting objects. 1992.
[18] BURSTALL, R.; GOGUEN, J. Putting theories together. IJCAI, 1977.
[19] HEBERT, J.-P.; SIMMONS, H. Semirings and Their Applications. Springer, 2009.
[20] KUICH, W.; SALOMAA, A. Semirings, Automata, Languages. Springer, 1986.
[21] CUNINGHAME-GREEN, R. Minimax Algebra. Springer, 1979.
[22] EILENBERG, S. Automata, Languages and Machines. Academic Press, 1974–1976.

[23] HOPCROFT, J.; ULLMAN, J.; MOTWANI, R. Automata Theory. 2001.
[24] SCOTT, D. Outline of a mathematical theory of computation. 1970.
[25] SCOTT, D.; STRACHEY, C. Mathematical semantics. 1971.
[26] PLOTKIN, G. LCF as a programming language. TCS, 1977.
[27] ТОШКОВ, А. Дефиниране на нов универсален математически агрегатор. КНК, 2025.
[28] ТОШКОВ, А. Свойства на универсалния агрегатор $A_{\circ *}$. КНК, 2025.
[29] KNUTH, D. The Art of Computer Programming. 1997.
[30] PAPADIMITRIOU, C. Computational Complexity. 1994.
[31] GRÄTZER, G. Universal Algebra. Springer, 2008.
[32] DAVEY, B.; PRIESTLEY, H. Lattices and Order. Cambridge, 2002.
[33] AHO, A.; SETHI, R.; ULLMAN, J. Compilers. 1986.
[34] LANG, S. Algebra. Springer, 2002.
[35] HALMOS, P. Naive Set Theory. Springer, 1960.
[36] BOURBAKI, N. Algebra I. Springer, 1989.
[37] BOURBAKI, N. Theory of Sets. 2004.