الثقافة المعلوماتية

مجلة دورية تصدرها الجمعية العلمية السورية للمعلوماتية

السنة الثالثة والعشرون – العدد السابع والخمسون– آذار "مارس"/حزيران "يونية" 2019

ملف العدد: قضايا في التطبيقات الحاسوبية

مخاطر الأتمتة: منظور النظام الشامل التحذيري لمستقبلنا السَيْبَري

أمَّا وقد صار بإمكاننا الكتابة تزامنيًّا، فكيف نستثمر ذلك لمصلحتنا؟

كل شيء عن الصور

التفكير العميق لتوليدِ كلامٍ أجود

الأبحاث الأخرى

مستقبل أنصاف النواقل

الحوسبة اللامتجانسة: وجدت لتبقى

تعزيز الإبداع بالحوسبة

البرمجة القابلة للتعلُّم: الكتل وما بعدها

البروفسكيتات تُعزِّز إمكانات الخلايا الشمسية

قائمة المصطلحات

الثقافة المعلوماتية

مجلة دورية تصدرها الجمعية العلمية السورية للمعلوماتية

السنة الثالثة والعشرون – العدد السابع والخمسون– آذار "مارس"/حزيران "يونية" 2019

رئيس التحرير:

الدكتور موفق دعبول

هيئة التحرير:

الدكتور سعد الله آغا القلعة

الدكتور سامح جزماتي

الدكتور نزار الحافظ

الدكتور راكان رزوق

الدكتور حسان ريشة

الدكتور عماد الصابوني

الدكتورة ندى غنيم

الدكتور منصور فرح

الدكتور محمد مراياتي

أمينة التحرير:

هيفاء باكير

للمراسلات:

الجمعية العلمية السورية للمعلوماتية

مجلة الثقافة المعلوماتية

دمشق – البرامكة– خلف كلية الفنون الجميلة

ص.ب. 33492 – سورية

هاتف: 0394 215 11 963+

بريد إلكتروني: nzhafez@scs-net.org

تنسيق المقالات وإعداد المجلة للطباعة: **الدكتور نزار الحافظ**

الإخراج الفني والمعالجة: **مركز الفوال للفرز والمونتاج الإلكتروني**

عزيزي القارئ،

اخترنا لهذا العدد أربعة بحوث تحت العنوان **قضايا في التطبيقات الحاسوبية**، هي: " *مخاطر الأتمتة: منظور النظام الشامل التحذيري لمستقبلنا السَيْبَري* "، " *أمَّا وقد صار بإمكاننا الكتابة تزامنيًّا، فكيف نستثمر ذلك لمصلحتنا؟* "، " *كل شيء عن الصور* "، " *التفكير العميق لتوليدِ كلامٍ أجود* ".

يلقي **البحث الأول** (مخاطر الأتمتة: منظور النظام الشامل التحذيري لمستقبلنا السَيْبَري) الضوء على مخاطر استعمال الحاسوب للأتمتة الشاملة للمنظومات، وأسبابها، والتحديات التي تواجه التقدم في هذا الميدان. ويتفحص منهجياتٍ مرغوبةً اقتصاديّاً، لكنها ذات مخاطر محتملة كبيرة وذلك بحسب التطبيق. ويدرس المقال التطبيقات الآتية، سلامة الملاحة الجوية وأمانها، والعربات الذاتية القيادة ونصف المؤتمتة، والطرق السريعة المؤتمتة، وإنترنت الأشياء، والحوسبة السحابية، والتخزين السحابي.

يستكشف **البحث الثاني** (أمَّا وقد صار بإمكاننا الكتابة تزامنيًّا، فكيف نستثمر ذلك لمصلحتنا؟) آليةَ الكتابة المتواقتة بواسطة عدة أشخاص على وثيقة واحدة، ويستعرض عدة روايات تبيّن مدى قدرتها الإنتاجية. ويناقش مدى الدعم التقاني المتاح لجعل هذه القدرة أشد فاعلية.

يتناول **البحث الثالث** (كل شيء عن الصور) الأهميةَ المتزايدة لتعرّف الصور بالحاسوب، والتقانات المعززة لها، مثل الشبكات العصبونية وخوارزميات التعلم الآلي ووحدات المعالجة البيانية، التي بدأت تساعد على تحليل تدفقات الصور الهائلة في مجالات علمية مختلفة.

يناقش **البحث الرابع** (التفكير العميق لتوليدِ كلامٍ أجود) مسألةَ توليد الكلام المنطوق بالحاسوب وجعله كالكلام الطبيعيّ من حيث نطقه نطقًا صحيحًا والانسيابية من كلمة الى الكلمة التالية، وتقنيات تركيب الكلام بالحاسوب، وجهود الباحثين في سبيل تحسين الكلام المنطوق.

واخترنا أيضاً لهذا العدد خمسة بحوث متنوعة المواضيع مثيرة للاهتمام هي على الترتيب: " *مستقبل أنصاف النواقل* "، " *الحوسبة اللامتجانسة: وجدت لتبقى* "، " *تعزيز الإبداع بالحوسبة* "، " *البرمجة القابلة للتعلُّم: الكتل وما بعدها* "، " *البروفسكيتات تُعزِّز إمكانات الخلايا الشمسية* ".

يَدرس **أول هذه البحوث** (مستقبل أنصاف النواقل) واقعَ صناعة أنصاف النواقل التي أصبحت في حالةٍ من التغير المستمر بسبب اقتراب أجل قانون مور الشهير، ويستشرف مستقبلَها وخياراتِ المنهجيات المتاحة لتصميم الدارات المتكاملة للمعالجات لحوسبة المستقبل.

يَستعرض **ثانيها** (الحوسبة اللامتجانسة: وجدت لتبقى) ماهيةَ الحوسبة اللامتجانسة، وفائدة التنوع من عقد الحوسبة، ويعطي إجابات لبعض الأسئلة: لماذا ستصبح الحوسبة اللامتجانسة معيارًا، وكيف نتعامل معها برمجيًّا وعتاديًّا. يطرح المقال أيضًا وجهات نظر مختلفة حول أسئلة أخرى.

يتناول **ثالثها** (تعزيز الإبداع بالحوسبة) لغةَ سكراتش مثالًا لأداة حوسبةٍ تمكِّن من إظهار الإبداع لدى الطلاب، ويتناول مدى تأثيرها في التفكير الإبداعي لديهم، وبعضَ أدوات هذا التفكير وتداخلاتها مع علوم الحاسوب.

يفصّل **رابعها** (البرمجة القابلة للتعلُّم: الكتل وما بعدها) البرمجةَ المعتمدة على الكتل، واللغات التي تدعم هذه النمط من البرمجة، وتأثير الكتل في قابلية تعلُّم البرمجة. يناقش البحث فعالية لغات الكتل، والميزاتِ الرئيسية للغات الكتل وارتباطها بالتعلم، وتطبيقات الكتل في المجالات الجديدة، والأدوات التي تمكِّن من إنشاءِ لغةِ كتل خاصة بالمبرمج.

أخيراً، يَكشف **خامسها** (البروفسكيتات تُعزِّز إمكانات الخلايا الشمسية) التقدمَ السريعَ الحاصل في التقانة الفوتوفولطائية باستعمال مواد تسمى البروفسكيتات، التي تساعد على تصنيع خلايا شمسية متقدمة تفوق الخلايا الشمسية التقليدية في المردود. ويناقش تحديات تصنيع هذه الخلايا على نطاق واسع.

يتضمن هذا العدد أيضًا قائمةً بأهم المصطلحات (إنكليزي – عربي) الواردة في مقالات هذا العدد.

أخي القارئ،

في الختام، نتمى أن تجد في مواضيع هذا العدد الفائدة المرجوة، ونأمُل أن تتواصل معنا بإرسال ملاحظاتك ومقترحاتك إلينا ...

وإلى اللقاء معك في العدد القادم.

رئيس التحرير

الدكتور موفق دعبول

المحتويات

ملف العدد: قضايا في التطبيقات الحاسوبية

مخاطر الأتمتة: منظور النظام الشامل التحذيري لمستقبلنا السَيْبَري 9

أمَّا وقد صار بإمكاننا الكتابة تزامنيًّا، فكيف نستثمر ذلك لمصلحتنا؟..... 19

كل شيء عن الصور 35

التفكير العميق لتوليدِ كلامٍ أجود..... 41

الأبحاث الأخرى

مستقبل أنصاف النواقل..... 47

الحوسبة اللامتجانسة: وجدت لتبقى..... 53

تعزيز الإبداع بالحوسبة 61

البرمجة القابلة للتعلُّم: الكتل وما بعدها 67

البروفسكيتات تُعزِّز إمكانات الخلايا الشمسية..... 87

قائمة المصطلحات

مخاطر الأتمتة: منظور النظام الشامل التحذيري لمستقبلنا السَيْبَري

RISKS OF AUTOMATION: A CAUTIONARY TOTAL-SYSTEM PERSPECTIVE OF OUR CYBERFUTURE*

Peter G. Neumann
ترجمة: د. خالد مصري
مراجعة: د. نزار الحافظ

لنقم بالأتمتة على نحوٍ صحيح، حيث تكون أمرًا لا مفرّ منه.

لا يزال العديد من المواضيع المُتعلقة بمخاطر الحاسوب التي جَرَت مناقشتها في أعمدة Inside Risks سابقة في هذه المجلة (CACM) حاضرةً حتى يومنا هذا. وعلى الأرجح، ستتفاقم شدةُ هذه المخاطر (والجديدة منها) أكثر مع تزويد النُظُم بالتشغيل المؤتمت أو نصف المؤتمت على نطاقٍ واسع. ستكون هناك حاجةٌ إلى قدرٍ أكبر بكثير من الثقة (trustworthiness) بالنظام الشامل، تشمل عتادياتٍ وبرمجيات نظام وتطبيقات أفضل، قادرةً على التسامح مع القيود البشرية والعوامل البيئية. ستستمر المخاطر بسبب عدم كفاية الموثوقية والأمن والخصوصية، إضافةً إلى عفوية المستعملين وعجزهم عموماً عن التعامل مع التقانة المُعقَّدة. نكتشف مرارًا وتكرارًا مخاطرَ غير متوقَّعةٍ ناتجةً عن ربط النُظُم الفرعية بعضها ببعض (مثلًا، انظر Beurdouche [2])، بسبب سلوكٍ غير متوقعٍ للنظام. هناك حاجةٌ ماسّة إلى العديد من الخطوات نحو الأمام في مجال البحوث وتطوير النُظُم وسهولة الاستعمال (user friendliness). كما نحتاج إلى أرضيةٍ مشتركةٍ وسطٍ بين المتفائلين (الذين يعتقدون بوجود أجوبةٍ سهلةٍ لِبعض المسائل المطروحة هنا) والمتشائمين (الذين لديهم شكوكٌ جدّية حول استعمالات الأتمتة والذكاء الصنعي المتزايدة– خصوصًا عندما يُعنى بهذه الاستعمالات أشخاصٌ لا يستوعبون التكنولوجيا استيعابًا جيدًا).

في هذا العمود، سأتفحص بعض المنهجيات التي قد تكون مرغوبةً اقتصاديّاً، لكنها ذات مخاطر محتملة كبيرة. يشمل ذلك سلامة الملاحة الجوية وأمانها، والعربات الذاتية القيادة ونصف المؤتمتة، وصولًا إلى الطرق السريعة المؤتمتة؛ ما يُسمى إنترنت الأشياء؛ والحوسبة السحابية والتخزين السحابي.

**نُشِر هذا البحث في مجلة Communications of the ACM، المجلد 59، العدد 10، تشرين الأول (أكتوبر) 2016، الصفحات 26 – 30.*

يجب أن تشمل معتمديةُ النظام الشامل متطلباتِ السلامة البشرية وأمانها وموثوقيتها ومناعتها ومرونتها حتى مع وجود بعض الصعوبات مثل الخطأ البشري والهجمات والبرمجيات الخبيثة. مع ذلك، نحتاج أيضًا إلى بنيانات نُظُم استباقية تُقلِّل بطبيعتها إلى أدنى حدٍّ مدى الثقة في المُكوّنات المختلفة، ومتطلباتٍ أخرى مثل المراقبة الشاملة والتدقيقية (auditability) والتشغيلية البينية (interoperability) والتوافقية (compatibility) والتجميعية (composability) القابلة للتنبؤ للمكوّنات، لتتيح الحصول على نُظُمٍ سهلة ذات مورّدين متعددين (multivendor). على سبيل المثال، تعرُّف الصوت والكلام وفهمهما، والترجمة الآلية والحوارات الذكية والأجوبة الآلية، هذه الاشياء لها بعض الإمكانات للإطاحة بالمعتمدية. يجب أن نعتمد أيضًا على نُظُمٍ وشبكات غير موثوقٍ بها أساسًا من نواحٍ مختلفة– وتصبح في بعض الأحيان أقل موثوقيةً بسبب الضعف البشري وسوء الاستعمال من الداخل والرغبات الحكومية المحتملة للنفاذ الاستثنائي الذي يتجاوز الأمن الهامشيّ بالفعل (مثلًا، انظر Abelson et al. [1]). بالمحصلة، نحتاج إضافةً إلى ذلك إلى نُظُمٍ تتسامح مع البشر (people tolerant). وقبل كل شيء، سنحتاج إلى تصعُّدية (scalability) التنجيزات لكل المتطلبات التي أشرنا إليها (سواءً كان التحكُّم الموضعي الفردي مطلوبًا أيضًا أم لا)، إضافةً إلى الرغبة الحتمية للترقيات (upgrades) من بُعْد في سبيل التصحيح السريع لنقاط ضعف النظام (vulnerabilities) ولإتاحة المجال للتطبيقات الجديدة. كل ذلك شاقٌ جدًا لأننا في الحقيقة نحاول أن نتطور تدريجيًا من منصات اليوم غير المستقرة. من ثَمّ، قد نتساءل عمّا إذا كانت بعض هذه الرغبات هي بالفعل أضغاث أحلامٍ لا يمكن تنجيزها وصيانتها واستعمالها مع ضمانٍ كافٍ بأن تكون المخاطر المتبقية مقبولة. لن يكون هناك نظامٌ مثالي على الإطلاق– خصوصًا تلك النُظُم التي تتطلب استقلالًا كبيرًا في التشغيل. مع ذلك، يبقى السؤال التالي قائمًا: ما هو الجيد إلى حدٍّ كافٍ؟؛ لا يمكن الإجابة عنه عمومًا، ويعود السبب في ذلك إلى حدٍّ بعيد إلى وجود إجاباتٍ مختلفةٍ تبعًا للتطبيقات المُحدَّدة.

سلامة إلكترونيات الطيران وأمنها

نحن بالفعل ندفع سلامة إلكترونيات الطيران وأمنها عمومًا إلى حدّ بعيد. إنّ تطوير عتاديات وبرمجيات نظام إلكترونيات طيران لا يمكن تخريبها من غير قصدٍ أو عن عَمْدٍ هو وضوحًا غير بديهي ومكلّف، ولكن هذا جزءٌ صغير من المشكلة الكلية فقط. صُمِّم في الأصل بمثابة برنامج الطيران الحر (Free-Flight)، مع وضع ذكاءٍ أشدّ بكثير في نُظُم التحكم داخل قمرة القيادة – بحيث يمكن أن يصبح المتحكمون الأرضيون في الحركة الجوية أقل أهميةً في الزمن الحقيقي. مثلًا، أصبحت نُظُم تجنب الاصطدام حاليًا راسخةً جيدًا وموثوقةً عمومًا. تحوَّل برنامج Free-Flight عمومًا الآن إلى برنامج الجيل الجديد للنظام الشامل (NextGen)، الذي سيدمج التحكم الأرضي والجوي معًا. مع ذلك، يمكن أن يُدخِل مفهومُ التحكُّم المؤتمت بشدة والموزَّع والآمن بين الطائرات المتقاربة– في السياق الأوسع للمطار وللجدولة البعيدة المدى في المسار، مع تحكُّمٍ شامل في حركة الطيران بالزمن الحقيقي (لا سيما في أوقات التأخر الناتج عن الطقس العاصف)– مخاطرَ متعددة محتملة. في حركة الطيران تلك، يمكن أن يخضع مراقبو الحركة الجوية والطيارون حاليًا لضغطٍ شديد في أوقات الازدحام الشديد وظروف الطقس الشاذة، عندئذٍ يكون تزويدهم بمساعدة أكثر ذكاءً بمعونة الحاسوب شيئًا مفيدًا، إذا كان توفير ذلك مؤكدًا. مثلًا، تَطوَّر طقم أدوات منح الشهادة DO-178C الجديد تطورًا كبيرًا، وهو أكثر تطورًا بكثير من أسلافه. وهو يقدِّم آمالًا كبيرة بأننا نستطيع زيادة سلامة الطيران وأمنه أكثر فأكثر.

إن سلامة إلكترونيات الطيران وأمنها هما بالطبع مصدرُ قلقٍ في كل أنحاء العالم، ليس على المستوى المحلي فقط، خصوصًا مع وجود دولٍ ولغاتٍ مختلفة– ومشاكل تتطلب معالجة الطوارئ. جرى إحراز تقدُّمٍ هائل في هذه الأمور، مع أنه

لا تزال هناك حالاتٌ حدّية[1] (corner cases) يمكنها تحدّي التحكُّم الملائم وتتطلب انتباهَ الطيَّار (واحتمال التدخل). مع ذلك، إن جعلنا معظم السيطرة في ايدي الأتمتةُ المتكاملة فيجب أن يشمل ذلك العتاديات والبرمجيات والاتصالات والطيارين الذين قد يكونون قادرين أو غير قادرين على تجاوز السيطرة بالحاسوب في حالات الطوارئ، والمراقبين الأرضيين ذوي الخبرة الممتازة والمُدرَّبين تدريبًا ممتازًا والدفاعات المضادة للمتسلِّلين المحتملين. كان التوجه أن توضع نُظُم الإعلام والترفيه (infotainment) على الشبكة المحلية ذاتها الخاصة بالتحكُّم في الطائرة، ربما من دون فصلٍ مناسب. إذن، يجب أن تُضمَّن منهجيةُ النظام الشامل تطويرًا أقوى لأمن الشبكة حتى يُضمن أن تكون نُظُم التحكُّم في الطيران معزولةً تمامًا عن وسائل الإعلام والترفيه والنُظُم.

تتضمن المشاكلُ الأخرى في منظور النظام الشامل سلامةَ المطار وأمنه، وتفتيشَ الركاب والصيانة الوقائية للطائرة في الوقت المناسب، والتدريب الشامل للطيَّار لاستباق الأحداث غير المتوقعة. نميل لوضع كل اهتمامنا في آليات دفاعٍ قليلة (وتشمل تلك التي لم تكن موجودة من قبل للتصدي لنقاط الضعف السابقة)؛ مع ذلك، فإن هذه الاستراتيجية غير قابلة للتطبيق عند وجود عددٍ كبيرٍ من نقاط الضعف.

من الضروري أيضًا أن نأخذ بالحسبان وجودَ طائراتٍ مسيَّرة دون طيَّار يُتحكَّم بها من بُعد تتقاسم الفضاء الجوي، وكلَّ المخاطر على سلامة الإنسان وخصوصيته أثناء الطيران وعلى الأرض. ستتطلب الطائرات المسيَّرة (التي يُتحكم بمعظمها يدويًّا أو بآلية شبه ذاتية حاليًّا، مع أنها يمكن أن تصبح ذاتية كليًّا في المستقبل) أمْنًا أفضل لتفادي الخسائر المماثلة لتلك التي أبدتها السيارات الحديثة– خصوصًا الطائرات المسيَّرة التي تحمل أسلحةً فتاكة.

السلامة اثناء الحركة في المَرْكبات المؤتمتة

تشمل مخاوف السلامة والأمن للنظام الشامل القدرة المُثْبَتة على اختراق عناصر التحكُّم في المركبات التقليدية– مثلًا، بواسطة مَنْفَذ الصيانة اللاسلكي أو بدلًا من ذلك بالتمكن من النفاذ إلى الشبكة الداخلية المحلية. يجب معالجة تلك المشاكل في المركبات الذاتية القيادة أو التي لها ميزات الأتمتة المرتفعة. لاحظ أن هناك تمييزًا بين السيارات الذاتية القيادة (مثلًا، سيارة غوغل Google، وإن كانت السيارة مع سائقٍ بديل أثناء مراحل الاختبار والتقويم الحالية، فالغاية أن تصبح مستقلةً كليًا)، ومساعَدة السائق المزيدة بالحاسوب (computer-augmented) (مثلًا، سيارة تسلا Tesla) التي تتجاوز أكثر الميزات شيوعًا مثل التحكُّم في الرحلات (cruise)، والوسائد الهوائية (airbags)، والمكابح المانعة للقفل (anti-lock braking)، ووقوف السيارات المتوازي (parallel parking)، وفيديو الرؤية الخلفية، وغيرها من التحسينات الحديثة للسلامة والراحة، ولكن هذا لا يرقى إلى مستوى التحكم الذاتي الكلي مع عدم القدرة على التدخل اليدوي.

كثيرًا ما يُصادف القاطنون في منطقة خليج كاليفورنيا سيارات غوغل الذاتية القيادة. إن مُعدَّلات الحوادث منخفضة جدًا حتى الآن، ويعود ذلك جزئيًا إلى أن المركبات مبرمجةٌ على المراقبة الشديدة لإشارات المرور ولظروف التغيرات البيئية لحالة الطريق –عادةً، مع سائقٍ بديل جاهزٍ لأخذ زمام المبادرة. (يوجد حالات يَصدم بها سائقون بشر مركبات غوغل من

[1] حالة حديّة: في الهندسة، تنطوي الحالة الحديّة على مسألة أو وضعية تحدث عندما تكون وسطاء التشغيل بعيدةً عن قيمها العادية– خصوصًا الحالة التي تحدث عندما تكون متحولات أو شروط محيطية متعددة عند أعلى مستَوَياتها في الوقت نفسه، مع أن كل قيمة وسيط تقع ضمن مجال قيمِهِ المسموح بها. (المصدر: ويكيبيديا). (المترجم)

الخلف– والسبب الرئيسي في ذلك هو البرمجة المُحافِظة لمركبات غوغل؛ يُعتقد أن السيارات التي تصدمها تكون قريبةً جدًا منها، مع كون السائقين لا يدركون الطبيعة المحافِظة لسيارة غوغل). يبدو أن الرغبةَ في الحدِّ بشدةٍ من معدلات الحوادث بواسطة أتمتة المركبات واقعيةٌ، مع أنه يرجح أن تكون هناك دائمًا حالاتٌ حرجة غير متوقعة. بالمناسبة، قامت غوغل بمراقبة بعض السائقين البُدلاء، واكتشفت أنهم لا يُوْلُون انتباهًا صارمًا كفايةً– ربما لأن أداء المركبات جيدٌ جدًا! مهما يكن الأمر، يبدو أن سجل مركبات غوغل الذاتية القيادة أفضل بكثير من سجل المركبات القديمة الطراز التي يقودها الإنسان. إذا أدركنا أن الأتمتة التي تتطور ما يزال العمل عليها جاريًا، فهناك أملٌ كبير.

لسوء الحظ، مات "سائق" إحدى سيارات تسلا S في 7 أيار 2016، في حادث اصطدام في فلوريدا عندما كانت سيارته في وضعية المساعدة الآلية[2]. هذا أول حادثٍ قاتل معروف أُعلِن عنه جرى لمركبةٍ يُتحكَّم بها آليًا. كان جوشوا براون (Joshua Brown) (وهو محارب قديم في البحرية أسَّس شركة الاستشارات التكنولوجية الخاصة به) في مقعد السائق، ولم تكن يداه تُمسِكان بالمقود، وكان مدافعًا صريحًا عن سلامة التحكم الآلي. (تشير التقارير الحديثة أنه كان يشاهد فيلمًا لهاري بوتر (Harry Potter). ينص المقال المذكور على أن " لا السائق الآلي ولا السائق البشري لاحظا الجانب الأبيض لمقطورة الجرار (التي انعطفت نحو اليسار أمام سيارة تسلا) وكانت السماء زاهية، لذلك لم يُضغَط على مكابح السيارة." يبدو أن التصادم أثار الشكوكَ فيما إذا كانت المركبات الذاتية القيادة عمومًا قادرة دائمًا على اكتشاف جميع المواقف المحتملة التي تُهدِّد الحياة. مع ذلك، لا يمكن أن يكون لوفاةٍ واحدة، بعد ملايين الأميال من القيادة التي سُجِّلت، أهميةٌ خاصة. هذا أفضل بكثير من قيادة الإنسان. مع أن التفاصيل تثير القلق، فحتى الأتمتة التي تبدو مثاليةً تؤدي إلى وقوع الحوادث والإصابات والوفاة؛ حتى مع الأتمتة، لا شيء مثاليٌ في الواقع.

علَّق كارل براور (Karl Brauer) (مُحلِّل في شركة Kelley Blue Book[3]): "هذا جزءٌ صغير من صرخة تنبيه. ربما كان الناس عدوانيين أكثر من اللازم في اتخاذ موقفٍ بأننا وصلنا تقريبًا، فهذه التكنولوجيا ستُطرح في السوق قريبًا جدًا، وربما نحتاج إلى إعادة تقويم ذلك." لكن إيلون موسك (Elon Musk) أشاد بسيارة تسلا طراز S وقال إنَّها" أفضل ربما من الإنسان حاليًا." وأيضًا أشار بيان شركة تسلا في 30 حزيران إلى أنّ قيادة سيارة تسلا طراز S مع تفعيل هذه التكنولوجيا (على أنها المختبِر بيتا! beta-tester) "يتطلب اعترافًا صريحًا بأن النظام هو تكنولوجيا جديدة."

إن رد الفعل الفوري إزاء "السائق الآلي" لسيارة تسلا هو أنه ينبغي ألاّ نطلق عليه اسم سائقًا آليًا، لأنه يتطلب صراحةً الانتباه المُستمر من الشخص الذي يَشغل مقعد السائق. برز هذا الاسم المغلوط مرارًا– خصوصًا في أعقاب الحوادث الأخيرة.

لم تكن سيارة تسلا التي أدَّت إلى وفاة براون مُجهزَّةً بـ "ليدار": ليزرٍ نبضي (كشف الضوء وتقدير المدى) LiDAR (Light Detection And Ranging)[4] ، وكانت تعتمد على كَمِرة من شركة Mobileye ورادار أمامي[5] (forward-facing radar). من الواضح أن هناك العديد من التحسينات التي يمكن إضافتها (مثل الليدار)– لا إلى عناصر التحكُّم في العربة

[2] انظر http://bit.ly/2aRzPqX

[3] Kelley Blue Book هي شركة متخصصة في تقويم السيارات والبحوث المتعلقة بها، تقع في إيرفين بكاليفورنيا، ومعترفٌ بها من قبل المستهلكين وصُنَّاع السيارات. (المترجم)

[4] الليدار تقنية تشبه الرادار تُستعمل لقياس المسافة عن هدفٍ، وذلك بإضاءته بضوءٍ ليزري وقياس الضوء المُنعكس عليه بواسطة مُحِس. عندها، يمكن استعمال المدة اللازمة لعودة ضوء الليزر في حساب المسافة. (المترجم)

[5] انظر http://bit.ly/297eo4D

فقط، بل تشمل أيضًا أتمتةُ المُحِسَّات والإشارات في الطرقات وخصوصًا عند التقاطعات الخطيرة ذاتها، واتخاذ حدودٍ مختلفة للسرعة ديناميكيًا في الظروف المناخية السيئة، وغير ذلك كثير.

في السادس من شهر تموز في عام 2016، ظهرت تقارير تفيد بأن سيارة تسلا X في وضعية "السائق الآلي" فقدت السيطرة في شارع رئيسي سريع مدفوع الأجر في ولاية بنسلفانيا، فارتدَّت على درابزين الحماية الإسمنتي وانقلبت على ظهرها؛ وأفادت التقارير أن الراكب في مقعد السائق لم يُوْلِ انتباهًا كافيًا، وقد أصيب[6].

أشار جون كوان (John Quain) [7] إلى وجود دليلٍ هام على أن السائق خلف المِقود قد لا يكون جاهزًا ليحلَّ محلَ السائق الآلي بسرعةٍ كافية لتجنب وقوع كارثة: " بيَّنت التجارب التي أجراها الباحثون في Virginia Tech العام الماضي والتي أيَّدتها إدارة السلامة الوطنية أن سائقي السيارات من المستوى 3 (حيث يمكن للسائق التخلي كليًا عن السيطرة على جميع وظائف السلامة الحساسة في ظروف معينة) يحتاجون إلى 17 ثانية وسطيًا (!!!) للاستجابة إلى وضعيات تولي السيطرة. في تلك المدة، تقطع سيارةٌ تسير بسرعة 65 ميل في الساعة مسافة 1621 قدم– أيْ مسافة تزيد عن خمسة ملاعب كرة قدم."[7]

بتعميم هذه الحالة، يبدو أن سؤالًا كبيرًا ينشأ فيما يتعلق بالمسؤولية القانونية– حيث تميل إجراءات التقاضي للبحث عن الموارد المالية الوفيرة. ولكن هناك العديد من القضايا هنا. ربما عندما تشتري مركبةً مؤتمتة، قد ينص العقد على أن السيارة في وضعية الاختبار، وأن الشركة الصانعة تتنصل من المسؤولية القانونية وتتخلى صراحةً عن مسؤوليتها. (هذا يشبه إلى حدٍّ ما مُقدمّي معظم نُظُم التشغيل الشائعة الذين يصرحون أنه ينبغي عدم استعمال هذه النُظُم للتطبيقات الحساسة– مع أنه على ما يبدو جرى تجاهل هذا التحذير على نطاقٍ واسع.) في تلك الحالة، يمكن أن يدَّعي محامو الشركة الصانعة بنجاح أن السائق كان مهمِلًا بثقته الزائدة أكثر من اللازم بنظام البرمجيات / العتاديات. تصبح القضايا القانونية أكثر تعقيدًا إذا أصرَّت دوريات الطرق السريعة على المراوغة لتكون قادرةً على إعادة توجيه المركبات أو إيقافها للتفتيش أو الاعتقال، الذي قد يؤدي بذاته إلى وقوع حادثٍ أو فعلٍ عنيف. وماذا يحدث فعليًا عندما تتصادم مركبتان أو أكثر بلا سائق ذاتيًّا التحكم كليًا؟ أو عندما تُختار مركبةٌ يُتحكَّم بها من بُعد لأغراضٍ شريرة؟ يوجد مخاطر عديدة جدًا لتعدادها هنا، وهناك حاجةٌ إلى مزيدٍ من البحث والتطوير والتقويم.

لنتأمَّل على وجه الخصوص، نشرتين وثيقتي الصلة بالموضوع كتبهما دون نورمان (Don Norman) [3,6] تستحقان القراءة بالفعل. ساهم دون نورمان ببعض الاقتباسات البليغة في مقالتي [4] في مجلة *Ubiquity* ACM في تموز 2016 عن هذا الموضوع. وهو يعتقد أن الأتمتة الجزئية كارثةٌ يُنتظر حدوثها، وأن الأتمتة الشاملة أساسيةٌ." إن التفكير خلاف ذلك تجاهلٌ لعقودٍ من البحوث الجادة في مجالي علم النفس والعوامل البشرية (ومجلس تكامل النظم البشرية التابع للأكاديمية الوطنية National Academy's Human Systems Integration board). ولا يوجد طريقةٌ للتغلب على ذلك. كلما كانت الأتمتة [الجزئية] أفضل، أصبحت أشدُّ خطورةً. يجب أن تكون الأتمتة شاملةً، وليس هذا المستوى 3 السخيف."

[6] انظر http://bit.ly/2aYNzBD

[7] Vlasic, B. and Boudette, N. نيويورك تايمز (1 تموز، 2016)، مع مشاركات متابعة Bill Vlasic في اليوم والأسبوع الذي تلاه؛ http://nyti.ms/2b2QC91

مع ذلك، فإن إدخال الأتمتة في أنشطةٍ تنظمها سلفًا المعايير التي لم يؤخذ بالحسبان في صياغتها الأتمتة والأمن يمكن أن يُعرِّض للمخاطر. ثم إنّ نقصَ الاستثمار في البنى الأساسية والطلبَ على التغيير التدريجي مع التوافق الرجوعي، قد يكون عائقًا أمام التقدم نحو السلامة والأمن.

أثناء كتابة هذا العمود، اطَّلَعت على أعمال مركز تحليل معلومات السيارات والمشاركة فيها (Automotive Information Sharing and Analysis Center) (Auto-ISAC) (الذي جمَّع مجموعَة من أفضل الممارسات) وقمة الأمن السيبيري الشامل للسيارات في بيلنغتون (Billington Global Automotive Cybersecurity Summit) (التي عقدت اجتماعها الافتتاحي في 22 تموز 2016). يبدو أن هذه الجهود تُكِّرر قلقي من وجوب مراعاة السلامة والأمن معًا في كل مراحل صناعة السيارات. بالفعل، هم يزعمون أنهم يفعلون ذلك دون السعي لجعل الأمن ميزةً تنافسية للشركات الفردية، ولمعرفة ما يمكن معرفته من القطَاعات الأخرى، ولإتاحة السيارات المؤتمتة أتمتَّة شاملَة على أساس البيع بالتجزئة (بالمفرق) العادي في السنوات العشر القادمة[8].

الطرق السريعة المؤتمتة

يبدو أن تصوُّر أن تكون كل مركبةٍ على الطريق السريع مؤتمتَّة (دون الخوف من الحوادث أو الإحباط من الازدحام) ما زال بعيدًا إلى حدٍّ ما. سيعتمد ذلك في نهاية المطاف على التنسيق التعاوني الوثيق بين المركبات المتجاورة، إضافةً إلى المُساعِدات الآلية ونصف الآلية التي أشرنا إليها في الفقرة السابقة، والاتصالات الموثوقة بأجهزة السيطرة على المركبات المجاورة ومخاطر الطريق. إضافةً إلى ذلك، سيكون ضروريًا وجود نوعٍ من مراقبة حركة المرور في النظام الشامل، خصوصًا لاكتشاف الحوادث والاستجابة لها، وظروف المناخ القاسية والسيارات التي نفد وَقودها أو بطاريتها والإطارات المثقوبة، وغير ذلك. هناك قلقٌ آخر، وهو بالطبع إدخال المركبات القديمة (مع الحدِّ الأدنى من الاستقلالية والمراقبة في الزمن الحقيقي) في الخلطة، أو ربما العيش مع حلٍ أبسط– منع هذه المركبات القديمة من الطرق السريعة المؤتمتة وإجبارها على سلوك الطرق الخلفية. قد تكون مشاكل التحكُّم الثنائية البعد أقل تحدِّيًا بقليل من مشاكل التحكُّم في طيران الطائرات الثلاثي الأبعاد، ولكنها مع ذلك مهمةٌ، خصوصًا في حالات الطوارئ المحتملة. مع ذلك، فإن الفواصل بين الأجسام المتحركة ومدةَ رد فعل الإنسان مقابل مدة رد الفعل المؤتمتة، وضمانَ المخاطر كل ذلك يختلف اختلافًا كبيرًا بين الطائرات في الجو والمركبات على الأرض. بطريقةٍ ما، فإن مشكلة الأتمتة في الطيران أبسط من مشكلة أتمتة السيارات. إنّ تعرُف الأشكال لمجموعةٍ متنوعة من الأجسام في خلفياتٍ مُربكة هامٌ في السيارات؛ أمّا في الطائرات، فعلى المرء ببساطة اكتشاف وجود الجسم– لأنّ الهوية الدقيقة لا تهمُّ كثيرًا (إلّا في حالة القتال). فضلًا على ذلك، قد يلزم أن تكون الاستجابة في القيادة من مرتبة أجزاءٍ من الثانية، في حين يُقاس الزمن اللازم في الطيران بالدقائق عادةً– أو حتى بالساعات في حالة التخطيط البعيد المدى لمنع الازدحام.

يُطبَّق مفهوم النظام الشامل تطبيقًا تامًا على الطرق السريعة المؤتمتة، إذ يجب مكاملةُ العديد من المسائل. يمكن ربط كل المحيط بمُحِسَّات وأجهزة قياس يمكنها التفاعل مع النُظُم الفرعية الفردية– مُعْلِمةً لها على نحوٍ ملائم في الزمن الحقيقي.

[8] "'Gene' تدحرج سيارة تسلا طراز X بعد اصطدامها بالحاجز المنصف الإسمنتي، وادعاء السائق أن السائق الآلي كان مفعَّلًا" (6 تموز 2016)؛ http://bit.ly/2bcr9KM وأخبار الوكالة الفرنسية للأنباء AFP، "اصطدام تسلا: انقلاب سيارة تسلا طراز X وهي في وضعية السائق الآلي، كما يقول السائق"؛ http:/bit.ly/2aMWmT0

سيولّد ذلك العديد من المشاكل المعقدة المتشابكة في النظام، التي تتطلب حلولًا قابلةً للتوسع وتتفادى الاستهلاك المُفرِط للطاقة.

نتيجةً لذلك، من المُرجَّح أن تكون القضايا القانونية والمسؤولية والخصوصية وغيرها من القضايا المذكورة هنا للمركبات المؤتمتة، أكثر تعقيدًا عند تطبيقها على التحكُّم الموزَّع في المركبات المؤتمتة ونصف المؤتمتة على الطرق السريعة المؤتمتة، حتى لو لم تكن مختلطة مع المركبات اليدوية التقليدية.

إنترنت الأشياء

في إنترنت الأشياء (IoT) يمكن أن يكون لكل شيء يمكن تخيله تقريبًا وجود على الإنترنت. لذا، يجب اعتبار إنترنت الأشياء في سياق النقاش السابق– لا سيما فيما يتعلق بتلك الأشياء التي يمكن النفاذ إليها مباشرةً على الإنترنت حاليًا. يمكن أن تكون التجهيزات آليةً تمامًا أو تعمل كليًا بتحكُّمٍ من الإنسان (ولكن مع مراقبة من بُعد)، أو مرةً أخرى بين هذا وذاك. سيكون بعضها قابلًا للتحكُّم فيه من بُعد، أو بدلًا من ذلك يمكن النفاذ إليه من الإنترنت. (يبدو هذا دعوةً مفتوحة للتلاعب غير المرغوب فيه، ولانتهاكات الاجتياح للخصوصية. مع ذلك، من المنطقي، أن يكون العديد من هذه الأشياء (Things) مخفيًا خلف جدار الحماية على الأرجح، لكن مع ذلك لا يزال من الممكن النفاذ إليها من بعد (مثلًا، بواسطة SSH[9]). إذا جرى البحث عن أرخص الحلول، فقد لا يكون هناك جدار حماية، وسيتطلب كل شيء بيئة حمايةٍ خاصةً به. وبخلاف ذلك، ستنمو الشبكة المظلمة[10] ("Dark Net") الموجودة على الإنترنت (التي لا يمكن البحث فيها) نموًا كبيرًا لاستيعاب كل الأشياء التي قد تكون مخفيةً خلف جدران الحماية التي يُفترض أنها آمنة. قد يؤدي ذلك أيضًا إلى تطوير جدران حماية قابلة للاختراق لتوفير المراقبة الحكومية في بعض الدول، والتي قد تفسح المجال لسوء الاستعمال من قبل آخرين أيضًا. نظرًا لنقاط ضعف جدران الحماية الحالية والحواسيب المكتبية والتجهيزات المحمولة، ستكون هناك حاجةٌ إلى حمايةٍ أفضل بكثير، حتى في النُظُم الصغيرة، في الأشياء الصغيرة والأشياء التي تبدو غير هامة، ولكن خصوصًا في جدران الحماية والموجِّهات (routers) الداخلية. بالفعل، ربما توفِّر هذه الأشياء (Things) التي تبدو غير هامة النفاذ لأشياء أخرى– بسبب احتمال النفاذ الكلي غير المقيَّد ضمن شبكة الأشياء (Things) وراء جدار الحماية.

إن قضايا الخصوصية غامضةٌ إلى حدٍّ ما. فمثلًا، حَكَم قاضيٌ فيدرالي في المنطقة الشرقية من فرجينيا أنه لا ينبغي لمستعمل أيّ حاسوبٍ موصول بالإنترنت أن يتوقع وجود خصوصية، لأن أمن الحاسوب غير فعَّال في إيقاف المخترقين (hackers). في 23 حزيران من عام 2016، جاء الحُكْم في واحدةٍ من القضايا العديدة الناتجة عن تسلُّل مكتب التحقيقات الفيدرالي FBI إلى خدمة PlayPen، وهي خدمةٌ مخفية في شبكة Tor كانت تقوم بدور مجمِّع لاستغلال الأطفال، والملاحقة القضائية اللاحقة لمئات الأفراد (يبدو حُكْم القاضي متعارضًا مع أحكامٍ أخرى، وهناك احتمال كبير لاستئنافه.) ولتعرُّف

[9] Secure Shell هو بروتوكول تعمية للشبكات، لتشغيل خدمات الشبكة بشكلٍ آمن في شبكة غير آمنة. (المترجم)

[10] الشبكة المظلمة مصطلحُ عريض (umbrella term) يصف أجزاء الإنترنت غير المفتوحة للجمهور قصدًا أو الشبكات المخفية التي يتراكب بنيانها مع بنيان الإنترنت. يترافق هذا المصطلح غالبًا مع الجزء المُعمى من الإنترنت المُسمى Tor network حيث يحدث الاتجار غير المشروع مثل سوق المخدرات سيء السمعة المُسمى Silk Road. (المصدر: ويكيبيديا) (المترجم)

المشتبه بهم، سيطر FBI على خدمة PlayPen مدة أسبوعين واستعمل برنامج تحقيق شبكي يُنفَّذ على حواسيب الزائرين لتعرُّف عناوينهم على الإنترنت[11].

قد نشك حاليًا في أنّ إنترنت الأشياء هي إلى حدٍّ بعيد فرصةٌ تسويقية للشركات، حيث تسعى كل شركةٍ لأن يكون لها نهجٌ صالح. مع ذلك، يبدو أيضًا أنّه لا يوجد هذا النهج– على الأقل حاليًا، وأنه يمكنك أن تتوقع وجودَ الكثير ممن يبيعون بضائع مزورة.

السُحب

إن الحوسبة السحابية والخَزْن السحابي منطقيان جدًا في الكثير من البيئات العملياتية. تبدو هذه الموارد لمعظم المستعملين ذاتية العمل (autonomous)، بمُدخلاتٍ بشرية ومخرجاتٍ يولّدها الحاسوب. مع ذلك، فإنها تثير الكثير من القضايا التي تتعلق بالوثوق بالسحب والشبكات، وبمن نثق أو بمن نحتاج إلى أن نثق. نذكر من الأمثلة على ما قد يكون شائكًا هنا على وجه الخصوص التعميةَ وإدارة المفاتيح، والنفاذ الاستثنائي لإنفاذ القانون، والصيانة والإصلاح عندما يحدث خطأٌ أساسي (مثلًا، انقطاع التيار الكهربائي أو وجود تهديدات). في آخر هذه المخاوف، أين تجد (أو أين يجد مزوِّد السحاب) سريعًا مديري نظام متمرسين ملائمين في حالة الأزمات؟ قد يكون معظم هذه القضايا بالكلية خارج سيطرة جماهير المستعملين.

المراقبة

يَجعل تقرير Keys Under Doormats [1] (المفاتيح تحت ممسحة الباب) الحجةَ التقنية التي تُخفِّض الأمن بغيةَ تبسيط وظيفة إنفاذ القانون فكرةً سيئة جداً: فهي مثلًا، سوف تفتح مجالًا هائلًا لنقاط الضعف المحتملة للاستغلال، وستدفع بلا شك مُزوِّدي النُظُم المحليين وزبائنهم المحليين في دولٍ مختلفة نحو إيجاد مصادر أخرى لنُظُمٍ آمنة. أيَّد العديد من كبار المسؤولين الحكوميين السابقين في الولايات المتحدة استنتاجات هذا التقرير.

يجب أن تترافق أيّة محاولة تطويرٍ للنُظُم الذاتية (autonomous) بمراقبة مُشدَّدة لكي تضمن عمل النُظُم على النحو الصحيح. بالنيتجة، يجب التصدي لتحدّيات تطوير مراقبةٍ ليست فقط جديرةً بالثقة وغير قابلة للتخريب وواعية للخصوصية، ولكن أيضًا ذات قيمة للتحليل الجنائي. ستُضاف مخاطر تعرّض الأمن المخفَّض للاختراق من طرف غير المراقبين ذوي الامتياز المفترضين (وفيهم المطلِّعون ذوو الامتياز) إلى حقيقة أنه يمكن أن تتعرض السيارات والتجهيزات الأخرى للاختراق من بُعد. بالنتيجة، يبدو أنّ متطلبات النُظُم الذاتية القابلة للمراقبة التي لا يمكن للآخرين المساس بها فكرةٌ تجمع بين نقيضين، أو ربما ذات صعوبة عَوْدية (recursively) – ذلك لأنها يمكن أن تتطلب نُظُمًا أكثر أمانًا بكثير في المقام الأول!

العلاج

يمكن معالجة بعض هذه المسائل (باستثناء "المراقبة التي لا يمكن المساس بها") بوجود عتاديات تُعزِّز التحكُّم في النفاذ بالغة الدقة (fine-grained) بالترافق مع الافتراض (virtualization) المكفول العتاديات، وتقسيم البرمجيات إلى أجزاء مستقلة (compartmentalization) قابلة للتصعُّد التي قد تكون أقل موثوقيةً. مثلًا، يجب ألّا تسمح التجهيزات المحمولة والحواسيب

[11] انظر http://nyti.ms/2aHGExM

المحمولة (الحضنية) للتطبيقات بالنفاذ غير المقيَّد إلى قوائم الاتصال والتطبيقات الأخرى دون إذنٍ صريح. ستكون العتاديات التي تساعد على تعزيز خصائص الأمن الصارمة مفيدةً جدًا. بالمثل، ستتطلب إنترنت الأشياء جدران حماية وشبكات محلية آمنة على محمل الجدّ، مع نُظُم فرعية تتناسب تكلفتها وتعقيدها مع حساسية الأشياء (Things). يمكن أن يؤدي التقدُّم في الطرائق الصورية دورًا أيضًا في زيادة ضمان الثقة بالعتاديات والبرمجيات لهذه النُظُم، ويشمل ذلك الاختبار والتقويم المبنِيَّيْن على أسس صوْرية. يمكن الاطلاع على بنيان نظام[12] CHERI [8] مثالًا على ما قد يكون ممكنًا مع تصميم عتاديات خالية من القيود مع أبدالٍ لنظام التشغيل ومُترجِمات قادرة على الاستفادة من مزايا العتاديات.

الاستنتاجات

يرغب مروِّجو النُظُم المعتمدة على الحاسوب الحديثة في تحقيق قفزاتٍ كبيرة للأمام باستعمال الأتمتة والمساعَدة المؤتمَتة في الزمن الحقيقي– في بعض الحالات وضْع إصداراتِ اختبار بيتا قبل الأوان. نحتاج إلى نُظُمٍ متعلقة بالحاسوب موثوقةً أكثر بكثير مما لدينا الآن، خصوصًا للاستعمال في النُظُم الحساسة. نحتاج أيضًا إلى متطلباتٍ أكثر صرامةً للنظام الشامل وللبنيان الإجمالي للنظام، نحتاج إلى هندسة تطوير أجود وإلى اختبار وتقويم للنظام الشامل، وربما فوق كل شيء إلى وعيٍ وفهمٍ استباقيين للمخاطر لأجل الزبائن المحتملين. إذا كنا روتينيًا سنملك نُظُمًا مؤتمةً كليًا –أو حتى نُظُمًا مؤتمةً جزئيًا تتطلب تدخلًا بشريًا فوريًا في بعض الحالات– فيجب أن يكون لدينا بحثٌ وتطوير للنظام أكثر تقدمًا بكثير، إضافةً إلى تعليمٍ يتعلق بالمخاطر المحتملة وكيفية التعامل معها عند حدوثها. يجب توسيع شمول القول المأثور القديم "ليكن المُشتري حذرًا" ليشمل المستعملين أيضًا.

المَراجع

[1] Abelson, H. et al. *Journal of Cybersecurity 1*, 1 (Nov. 2015), Oxford University Press; http://bit.ly/2bcj1dr

[2] Beurdouche, B. et al. A messy state of the union: Taming the composite state machines of TLS. In *Proceedings of the 36th IEEE Symposium on Security and Privacy*, San Jose, CA, May 18–20, 2015; http://bit.ly/2bndXGz

[3] Casner, S.M., Hutchinson, E.L., and Norman, D. The challenges of partially automated driving: Car automation promises to free our hands from the steering wheel, but might demand more from our minds. *Commun. ACM 59*, 5 (May 2016).

[4] Neumann, P.G. Automated car woes—Whoa there! *ACM Ubiquity*, July 2016; http://bit.ly/2aYKDoT

[5] Neumann, P.G. *Computer-Related Risks.* Addison-Wesley and ACM Press, 1995.

[6] Norman, D.A. The human side of automation. *Road Vehicle Automation 2*, Springer, 2015.

[7] Quain, J.B. The autonomous car vs. human nature, a driver behind the wheel may not be ready to take it. *The New York Times* (July 8, 2016).

[8] Watson, R.N.M. et al. CHERI: A hybrid capability system architecture for scalable software compartmentalization. In *Proceedings of the 37th IEEE Symposium on Security and Privacy* (San Jose, CA, May 18–20, 2015).

[12] نظام CHERI: مختصر (Capability Hardware Enhanced RISC Instruction) هو نظام يوفر فحصًا دقيقًا إضافةً إلى التقسيم إلى أجزاء (compartmentalization) تديره البرمجيات ويجري تطبيقه في العتاديات. (المترجم)

أمَّا وقد صار بإمكاننا الكتابة *تزامنيًّا*، فكيف نستثمر ذلك لمصلحتنا؟

NOW THAT WE CAN WRITE *SIMULTANEOUSLY*, HOW DO WE USE THAT TO OUR ADVANTAGE? *

Ricardo Olenewa, Gary M. Olson, Judith S. Olson, Daniel M. Russel
ترجمة: أ. سعيد الأسعد
مراجعة: أ. مروان البواب

تتيح معالِجاتُ النصوصِ اليومَ لعدَّة مؤلِّفين إمكانَ العملِ على وثيقةٍ واحدةٍ في وقتٍ واحد. لكن ماذا عساهم أن يفعلوا في الواقع؟

تتيح معالِجاتُ النصوصِ الحديثةُ، مثل: Microsoft Word OneDrive وSharePoint وGoogle Docs، للناس العملَ على الوثيقة نفسِها في الوقت نفسِه. وفي حين عُرِضَتْ في مختبَرات البحث، ولبعض الوقت، نُظُمٌ تسمح بالكتابة المتواقتة (simultaneous writing)، فإنَّ أمثال هذه النُّظُم لم تصبح متاحةً للاستعمالِ على نطاقٍ تجاريٍّ واسع إلّا إلى عهدٍ قريبٍ نسبيًّا. فمعالِج النصوص Google Docs، على سبيل المثال، يمكِّن الأفرادَ أن يجتمعوا عند موضعٍ واحدٍ من الوثيقة، يضيفون إلى النص ويحذفون منه وينقلونه كما يشاؤون. وهو شائع الاستعمال على نطاقٍ واسع (فثمة مثلًا ما يزيد على 60 مليون مستعمِلٍ لتطبيقات غوغل في مضمار التربية وحدَه في أنحاء العالَم، ومليونَيْ مؤسسةٍ تجارية تستعمل هذه التطبيقات في العمل). ويرى الكثيرون في مزيَّة الكتابة المتواقتة ذخرًا نفيسًا.

وعندما أجرينا تحليلًا تفصيليًّا لنماذج كتاباتِ طَلَبَةٍ يتَّبعون دورةً جامعيةً متقدِّمةً في مجال إدارة المشروعات، لاحظنا أن 95% من الوثائق أظهرت نزعةً للكتابة المتواقتة، في حين أن ثلاث وثائق كُتِبت فعلًا كتابةً متواقتةً لا غير[9].

وقد خضعنا، نحن الأربعةَ أصحابَ هذا المقال، لدوراتٍ متعدِّدةٍ حَقَّقَتْ فيها الكتابةُ المتواقتةُ فوائدَ معتبَرة. وإذا كان بمقدورنا العملُ تزامنيًّا الآن، فكيف لنا أن نسخِّر تلك القدرةَ لجعل العمل أشدَّ فاعليةً؟ وما طبيعة ما يمكننا فعلُه في سياق الكتابة المتواقتة؟ ومتى يمكن أن نقول إننا *لا* نريد أن نكتب تزامنيًّا؟

**نُشِر هذا البحث في مجلة Communications of the ACM، المجلد 60، العدد 8، آب (أغسطس) 2017، الصفحات 36 – 43.*

للإجابة عن هذه الأسئلة طفقنا نجمعُ رواياتنا، ونصنِّفُها في مجموعات، ونثبتُ أنماطًا (patterns) تنتظمها. وتُروى كلٌّ منها بصوتِ واحدٍ منّا، إلا الرواية الأخيرة، التي تَشارَكْنا –أربعتُنا– في كتابتها، لأنها تُصَوِّر كيف ألَّفْنا *هذا* المقال. ثم إننا أثبتنا أوجهَ التشابه والاختلاف في رواياتنا، وأنشأنا –بجهدٍ جماعيٍّ موحَّد– النظامَ ذا الأنماط السِّتَّةِ والظاهرتَيْن الملحَقَتَيْن (epiphenomena). ونأمُل أن تكون الرواياتُ باعثًا للقرّاء على انتهاج سُبُلٍ للعمل جديدةٍ ومفيدة.

والتطبيق Docs من Google جديدٌ نسبيًّا؛ ومع أن بعضَ رواياتنا يشير إلى استعماله، فإنَّ عددًا منها يشتمل على نماذج بحثيَّةٍ أوَّلية (research prototypes)، أو منظوماتٍ تجاريةٍ قديمةٍ يرقى بعضُها إلى ثمانينيات القرن العشرين. إذن فلنَعْرض، قبل أن نشرع في تناول الروايات، وصفًا موجزًا لمزايا كلٍّ من هذه النُّظم وفقًا لترتيب إنشائها.

النُّظُم التي استعنّا بها

بيئة IDE(*): استُعمِل هذا النظام[10] أداةً لإنشاء محتوًى تعليميٍّ من طريق مساعدة المصمِّمين على تنظيم الموادِّ وإنشائها بسرعة، وقد بُنِيَ إضافةً إلى نظام الوسائط الترابطيّة البطاقيّة (Notecards hypermedia system)[5,6] واشتَمَلَ في المقام الأول على «بطاقات» ذات أنواعٍ مختلفةٍ من الروابط بين أجزاء النص. ومع أن نظام IDE لم يكن في الأصل مهيَّأً ليكون نظامًا تشاركيًّا لعدة أشخاص (multi-person collaborative system)، سرعان ما أدرك الناسُ إمكانَ استعماله باقتسام العمل فيما بين عددٍ من المؤلِّفين ابتغاء إنجاز العمل بسرعةٍ أكبر، وذلك بالعمل على بطاقاتٍ مختلفةٍ في وقتٍ واحد؛ فإذا كان أحدهم يعمل على بطاقةٍ بعينها، أُغلِقَتْ هذه البطاقةُ على الآخرين. ولئن كان هذا –من الناحية التقنية– ليس هو نمطَ التحرير المتواقت (simultaneous editing) الذي تتَّصف به الأدواتُ الأخرى، إنَّ ثمة دروسًا مفيدةً تُستخلَص من الواقعة التي استُعمِل فيها هذا النمطُ تحريرًا متواقتًا مُوازيًا (parallel simultaneous editing).

ShrEdit: أداةُ كتابةٍ تشاركيةٌ أُنشئت عام 1990[7,8]، إذ ضُيِّقَتْ وثيقةٌ مشترَكةٌ على مخدِّم، ونَفَذَ إليها الأفرادُ من آلات زبون موجودةٍ على الشبكة المحلّية نفسِها. وأتاح البنيانُ لعدة أفرادٍ الكتابةَ في الوثيقة ذاتِها في وقتٍ واحد. يُذكَر أنَّ لنظام ShrEdit «إقفالًا انتقائيًّا» (selection locking)، وهذا يعني أن أحدهم يمكنه أن يكتب تزامنيًّا مع آخرين ضمن حدود مِحْرَفٍ واحدٍ يفصل أحدَهم عن الآخر.

Aspects: منتَجٌ تجاريٌّ كان متاحًا في تسعينيات القرن الماضي[4]، وفيه –شأن ShrEdit– يستطيع الأفرادُ التحريرَ في وقتٍ واحدٍ ضمن حدود مِحْرَفٍ واحدٍ يفصل أحدَهم عن الآخر. وفي حين تَقَبَّلَ نظامُ ShrEdit النصوصَ فقط، تَقَبَّلَ Aspects وُرَيْقاتِ الجدولة (spreadsheets)، والرسومَ (drawings)، والعُروضَ (presentations)، إضافةً إلى النصوص، بما يشبه تطبيقات غوغل (Google Apps)، غير أنه يعمل على شبكةٍ محلّيةٍ ولا يتوفَّر إلا لحواسيب ماكنتوش الشخصية.

Centra Symposium: نظامٌ تجاريٌّ أتاح إجراءَ ائتمارٍ مسموعٍ ومرئيّ (audio and video conferencing)، إضافةً إلى التشارُك في غَرَض (object)، كوثيقةٍ أو عَرْض، ويَسمح للآخرين بتحرير الغَرَض المشترَك.

Docs من Google: نظامُ تحريرٍ تجاريٌّ، متعدِّدُ المستعملين، تشاركي الوثائق من غوغل، انطلقَ عام 2006. يمكِّن

(*) مختصر: Instructional Design Environment (بيئة التصميم التعليمي)

Docs عدَّةَ أفرادٍ من تحريرِ وثيقةٍ تحريرًا متواقتًا، ومعاينةِ تاريخ مراجعة الوثيقة وتعديلها، والتشاركِ مع مجموعةٍ خاصةٍ من الناس أو عامَّتهم على شبكة الإنترنت المفتوحة.

قوَّة الكتابة المتواقتة

ننتقل الآن إلى تناوُلِ رواياتِنا (stories) التي تُصَوِّر كيف لقوة الكتابة المتواقتة أن تُحْدِث بيئاتِ عملٍ أكثرَ إنتاجيَّةً إلى حدٍّ بعيد. وقد صُنِّفَت الرواياتُ في خمس مجموعات:

- كتابة وثائق موسَّعة (الروايات 1 – 3)
- كتابة وثائق قصيرةٍ تتَّصل بالواجبات الصَّفِّيَّة المدرسيَّة (الرواية 4)
- عرض وقائع الاجتماعات وإنشاؤها تشاركيًّا (الروايتان 5 – 6)
- التعليم بوساطة الوثائق المشترَكة (الرواية 7)
- كتابة هذه الوثيقة [المقال الذي بين أيدينا] (الرواية 8)

الرواية 1: طريقة التوثيق Doc Build-It باستعمال Google Docs، يرويها ريكاردو. لكي تكون مفيدًا للآخرين، عليك أن توثِّق برمجيّاتِك. وفي كثير من الأحيان يحتاج هذا التوثيق إلى إعادة كتابتها عدَّة مراتٍ من نقطة الصفر بسببٍ من عدم العثور على الأصل، أو عدم فهم هذا الأصل إن وُجِد. وممّا يؤسَف له أن تجد الكثيرين من مطوّري البرمجيّات يعافون التوثيقَ ويرغبون عنه. ومع أن الكُتّابَ التقنيّين، أمثالي، يستطيعون النهوضَ بهذا العمل، إلا أنهم قلّةٌ قليلةٌ حقًّا؛ إذ إن تفهُّمَ تقانةٍ ما وتدوينَ توثيقٍ يناسبها عملٌ شاقٌّ قد يستغرق سنوات. ومن ثَمَّ فلا يوثِّق عمليًّا من التقانات إلّا الأساسيُّ والمهمّ.

لحلِّ هذه المسألة، استَحدثتُ طريقةً للتوثيق سمَّيتُها Doc Build-It، تَصِفُ حَدَثًا يستغرق يومًا واحدًا، ويتمثَّل في اجتماع ثلَّةٍ صغيرةٍ من المهندسين لكتابة توثيقٍ متواقتٍ لجزئيَّةٍ معيَّنة من التقانة، باستعمال طريقتي في التوثيق Doc Build-It، وهي فعّاليةٌ مقيَّدةٌ مُعَدَّةٌ بعنايةٍ لتُمَكِّن المهندسين من إضفاء خبراتهم بطريقةٍ تبدو طبيعيةً لهم.

تتألف طريقةُ Doc Build-It من ثلاث مراحل: الإعداد (preparation)، والتركيب (أو التأليف) (composition)، والتحرير. ففي مرحلة الإعداد يلتقي الكاتبُ التقنيَّ الرئيسَ ليحصل منه على فكرةٍ عامةٍ وتفصيلية عن التقانةِ المرادِ توثيقُها. واستنادًا إلى ذلك ينشئ الكاتبُ صورةً مجملةً (outline) محتملةً للوثيقة النهائية. تتَّخذ هذه الصورةُ المجملةُ شكلَ مخطَّطٍ «حُبَيْبِيٍّ» (granular) مجزَّأً إلى نقاطٍ متميّزة، يحظى فيها كلُّ موضوعٍ بفكرتين أو ثلاث فِكَرٍ نقطيةٍ (bullet points) سديدةٍ تُفْصِح عن محتواه.

أما مرحلة التركيب فهي حَدَثٌ يستغرق يومًا واحدًا. يطلب الكاتبُ إلى التقنيّ الرئيسِ أن يدعو 3 – 5 من مهندسي الفريق إلى مكان الحدث، حيث يتوجَّه الكاتبُ أولًا بالطلب إلى المهندسين لتقديم شرحٍ شفهيٍّ عن التقانة بقصد وضعهم في الحالة الذهنية (mindset) للتعليم، الملائمة للتوثيق. يقوم الكاتبُ بعدئذٍ بتعديل المخطَّط المحتمل اعتمادًا على الشرح المقدَّم. وبموافقة الفريق عليه يدعو الكاتبُ المهندسين إلى تحمُّل مسؤولياتهم حيال الموضوعات التي يشعرون أنهم أكثرُ خبرةً فيها. وتقتصر مهمَّتُهم هنا على التقاط الفِكَر بالألفاظ التي وردت أثناء مرحلة الشرح. ثم إنهم ينصرفون جميعًا إلى الكتابة في وقتٍ واحدٍ في الوثيقة نفسِها، بحيث يستطيع الواحدُ منهم أن يرى الآخرين وهم يقدِّمون فقراتهم؛ ومن شأن ذلك أن يتيح لهم ضبطَ أساليبهم (مثل مستوى التفصيل)، وإثبات إحالات (cross-references)، وإجراء عمليات تدقيقٍ مزدوج (double check)

توخّيًا لمزيدٍ من الدقة.

يشجّع الكاتبُ المهندسين على عدم القلق لجهة سلامة الأسلوب في الكتابة، أو التهجئة، أو علامات الترقيم، أو قواعد النحو، أو التراكيب؛ فإسهامهم في الوثيقة يقتصر على إبداء مخزونهم المعرفي. تستغرق هذه الجلسةُ في العادة نحو ثلاث ساعاتٍ للموضوعات البسيطة نسبيًّا، وقد تمتد حتى سبع ساعاتٍ للموضوعات التي هي أكثر عمقًا.

بعد استكمال مرحلة التركيب، يقوم الكاتبُ التقنيُّ وحدَه بصقل الوثيقة وتهذيبها. ولمّا كانت عمليةُ التحرير تحتملُ وقوعَ أخطاءٍ دلاليَّة (semantic errors)، يقوم الكاتبُ بتوزيع نُسَخٍ من الوثيقة المحرَّرة لمراجعتها مع فريق المهندسين.

كان لطريقة Build-It في التوثيق أثرٌ كبيرٌ في الإنتاجية؛ فقد كانت تكلفةُ واقعةٍ واحدةٍ منها، أدَرْتُها بنفسي، أقلَّ عشرَ مراتٍ من أي طريقة توثيقٍ تقليدية، إذ بلغت تكلفتُها 1,900 دولار بدلًا من 18,000 دولار. وأَفْضَتْ أمثلةٌ أخرى للتوثيق بهذه الطريقة إلى نتائج مشابهة، وكان لها فوائد ثانويةٌ إضافيةٌ من قبيل تعرُّف عثراتٍ دقيقة (bugs) في التقانة. إن هذه الاكتشافات ممكنة، لأن الحدثَ يلفت نظرَ أكثرِ الناس خبرةً وأوسعِهم اطِّلاعًا في مبحثٍ معيَّن، ممَّن يركِّزون من كثبٍ على تصميم البرمجيات في سياق إنشاء وثيقةٍ ما.

رؤًى في الرواية. إن تنفيذ العمل في مراحلَ واضحةٍ رَسَمَ معالمَ الأدوار وبَسَّطَ العمليةَ برمَّتها. فالكاتبُ التقنيُّ أنشأ مخطَّطًا مُجمَلًا، ثم حُرِّرَ هذا المخطَّطُ المجمَلُ بعد الشرح والتفسير. وبعدئذٍ دَوَّنَ خبراءُ غايةَ ما استطاعوا تدوينَه من مخزونهم المعرفي، ثم قامَ الكاتبُ التقنيُّ بتهذيب هذه المعرفة المدوَّنة والتدقيق فيها، وجَرَت مراجعتُها التماسًا لمزيدٍ من الدقَّة. ومن الروايات الآتية ما يحمل مزيجًا مشابهًا من العمل المتواقت واللامتواقت.

الرواية 2: كتابة كتابٍ مدرسيٍّ باستعمال نظام IDE، يرويها دان. في صيف عام 1988، نَظَّمتُ فريقًا من عشرة طلبةٍ للدراسات العليا من مدرسة ستانفورد للتعليم، وذلك للمساعدة في وضع كتابٍ مدرسيٍّ في الجبر للمدارس الثانوية يحقِّق مطلبَ خبرةٍ في تصميم أنشطة الدورة†. وكان الغرضُ من ذلك كلِّه إخراجَ كتابٍ دراسيٍّ متكامل مع فريق عملٍ في غضون أقلَّ من عشرة أسابيع. وقد كَتَبَ الفريقُ النصَّ باستعمال نظام IDE.

أدّى اثنان من الطلبة العشرة وظيفةَ مصمِّمَيْن رئيسيَّيْن، فأنشأا مخطَّطًا إجماليًّا مبدئيًّا (initial outline)، وعَهِدا إلى كلِّ طالبٍ من الفريق بإنشاء فصلٍ من فصول الكتاب. ولكي «يكتب» الطالبُ (المؤلِّفُ) الفصلَ، كان عليه استحداثُ شبكةٍ من العُقَد (nodes) في نظام IDE، كلٌّ منها مرتبطةٌ بالعُقَد الأخرى، دلالةً على ماهية العُقَد الموصول إليها (linked-to nodes): هل هي مفاهيمُ مجرَّدة، أم تفاصيل فكرية، أم نماذج محسَّنةٌ عن نماذج سابقة، أم مسائل تطبيقية للتدريب، أم بِنًى يَعتمد بعضُها على بعض؟

في أثناء الكتابة، كان على المؤلِّفين تحديدُ مستلزماتٍ ربما يأخذون بها أثناء كتابتهم لمادّتهم. فمثلًا، يعتمد الفصلُ الذي يتناول الدوالَّ (التوابع) المثلَّثاتية (trigonometric functions) على تقديم مبرهنة فيثاغورَس (Pythagorean theorem) في فصلٍ سابقٍ تولّى كتابتَه مؤلِّفٌ آخر. من أجل مثل هذا الاعتماد كان من الضروري إنشاء رابطٍ صريحٍ يشير إلى أن هذا المفهوم (في فصلٍ ما) يعتمد على مفهومٍ آخر في فصلٍ آخر (يكون سابقًا عادةً).

† أعمال الطالب ما عدا الامتحانات. (المترجم)

يمكن بعد ذلك استقراءُ الشبكة الحاصلة بيانيًّا (بتتبُّع روابطَ من نوعٍ محدَّد) لإنتاج خَرْجٍ خَطِّيٍّ (linear output) وحيدٍ شبيهٍ بكتاب، يشمل المخطَّطاتِ والنصَّ من كلِّ عقدة.

كنّا نجتمع كلَّ يوم اثنين لمراجعة العمل المُنجَز طوال الأسبوع السابق، فيتبادل أعضاءُ الفريق الآراءَ ويناقشون قضايا الخلاف المفصليَّةَ ويجدون لها حلولًا، ويفسِّرون التداخلاتِ فيما بين أجزاء العمل. ولعلَّ مما كان يسهِّل –إلى حدٍّ ما– إيجاد الحلول استقلاليةَ كلِّ كاتبٍ بعمله تقريبًا (إلا عندما يتعيَّن عليهم تحريرُ عقدةٍ معًا بالجلوس جنبًا إلى جنب)، علمًا بأن أكثر القضايا طرحًا للحلِّ يتَّصل بالأسلوب ولهجة الخطاب واللغة وصوغ المفاهيم.

كذلك كنّا، قبل كلِّ اجتماعٍ أسبوعي، نقوم بطباعة البيان (graph) الذي يَعرض كاملَ العملِ المنجَزِ حتى تاريخه، وتعليقه على الجدار ليطَّلع عليه أفرادُ المجموعة جميعًا. يُظهِر هذا الملصَقُ (poster)، أسبوعًا فأسبوعًا، العملَ الذي أنجزه كلُّ عضوٍ من أعضاء الفريق، إضافةً إلى الروابط فيما بين الأجزاء ومكوِّناتها المختلفة. لم يكن البيانُ تقتصر فائدتُه على تحديد أين يمكن أن تنشأ المشكلات، بل كان أيضًا باعثًا قويًّا لأعضاء الفريق الآخرين على البقاء على اطِّلاعٍ تامٍّ على تطوُّرات الأمور.

وفي أثناء أيٍّ من تلك الأسابيع، كان بإمكان المؤلِّفين العملُ بصفةٍ مستقلةٍ على التوازي؛ فكانوا يكتبون العُقَدَ (nodes) الخاصةَ بهم، وينشئون ما يلزم من بِنًى أساسية (substructures). ولدى اكتشافِهم مستلزماتٍ جديدةً كانوا مسؤولين عن إنشاء تلك الروابط الصريحة فيما بين العُقَد حسب مقتضيات الحاجة. وكان بإمكان المؤلِّفين الآخرين، بطبيعة الحال، إنشاءُ ما يلزم من روابط جديدةٍ ضمن حدود نطاقاتهم الخاصة كذلك. وفي حال الروابط التقاطعية (cross-links)، كان المؤلِّفون أحيانًا يؤْثِرون العملَ معًا على نحوٍ متواقتٍ جنبًا إلى جنبٍ، ويقوم أحدهم بدور الكاتب لوقائع النقاش.

بعد ثمانية أسابيع اكتملَ المشروع – فجميع الفصول أصبحت تامَّةَ النصوص، وحُلَّت جميعُ النقاط التقاطعية الخلافية فيما بين الأجزاء. وقد أثمرت النسخةُ المطبوعةُ من العمل كتابًا دراسيًّا من 220 صفحة.

رؤًى في الرواية. أُلِّفَ الكتابُ في المقام الأول في جلساتٍ متوازيةٍ عَمِلَ فيها المؤلِّفون فرادى، وتَخَلَّلَتْها أوقاتٌ قصيرةٌ حاسمةٌ من العمل المتواقت عند بروز قضايا خلافٍ تحتاج إلى اتِّخاذ قرارات. أنشئ الكتابُ في غضون ثمانية أسابيع، مقارنةً بطريقة Build-It، التي مرَّت بعملٍ تمهيدي (مرحلة الإعداد)، ويومٍ طويلٍ حافلٍ من تفريغ المادة (dumping material)، وعدةِ أيامٍ من عمليات التهذيب. تجدر الإشارة إلى أن تطبيقَيْ Atlas O'Reilly (atlas.oreilly.com) و Book Sprints (www.booksprints.net) يدعمان إجرائيةً شبيهةً بهذه الرواية.

وهذه الروايةُ شبيهةٌ أيضًا برواية بويلشتورف (Boellstorf) وآخرين[2]، إذ يروي أربعةُ مؤلِّفين، بالتفصيل، واقعةَ إنشاء كتابهم المشترك الذي كُتِبَ بنظام Docs من غوغل. يعرض الكتابُ عمليات تناقُل المسؤوليات (توزيع المهام) واقتسام فقرات الكتابة. وفي وقتٍ لاحق استثمَر المؤلِّفون القدرةَ على الكتابة المتواقتة لتحديد شخص الكاتب ومكان الكتابة، والشروع في محادثةٍ تكون إما بالصوت وإما بمزيَّة محادثةٍ مرتبطةٍ بالصوت. وقالوا إنَّ هذا كان مفيدًا للتصحيح السريع والتشجيع في آنٍ معًا. وأخبروا أيضًا عن جلساتٍ كان فيها أحدُهم يكتب وآخر يُملي وثالثٌ يتتبَّعهم من كثبٍ لإجراء تنقيحاتٍ صغيرة، تحاكي ما سيأتي في الرواية 5.

الرواية 3: إنشاء تقرير لجنةٍ باستعمال أداة ShrEdit، ترويها جودي. دَرَجَ قسمُ علم الحاسوب في جامعةٍ كبرى

على إجراء مراجعةٍ خارجية كلَّ بضع سنوات. ولهذا الغرض تَفِدُ إلى الجامعة لجنةٌ استشاريةٌ، من خارج مِلاك الجامعة، مؤلَّفةٌ من نحو ثمانية أشخاص، لبضعة أيام لكي تنظر في أعمال الباحثين، وتراجع المنهاجَ الدراسي، وتدقِّق في إحصائيات القَبول وتحديد مستويات الطلبة، وما إلى ذلك. وجرت العادةُ أن تقوم اللجنةُ –بعد يومين من تقديم العروض والمحاضرات– بالتخطيط لطريقة صوغِ تقريرها في قالبٍ كتابيٍّ نهائي، واقتسام أعضائها الكتابةَ فيه، ومن ثَمَّ تلاوته على الآخرين والتعليق عليه بصورةٍ لامتزامنة (asynchronous) على مدى الأسابيع الأربعة القادمة.

وإذ لمسنا مدى فائدة نظام ShrEdit في أعمالنا الخاصة، قمنا بتنظيم شيءٍ شبيهٍ بطريقة التوثيق Doc Build-It، مع بعض الفروق المهمة. ودَعَوْنا اللجنةَ الاستشاريةَ إلى غرفةٍ خاصَّةٍ مزوَّدةٍ بعددٍ كبيرٍ من الحواسيب.

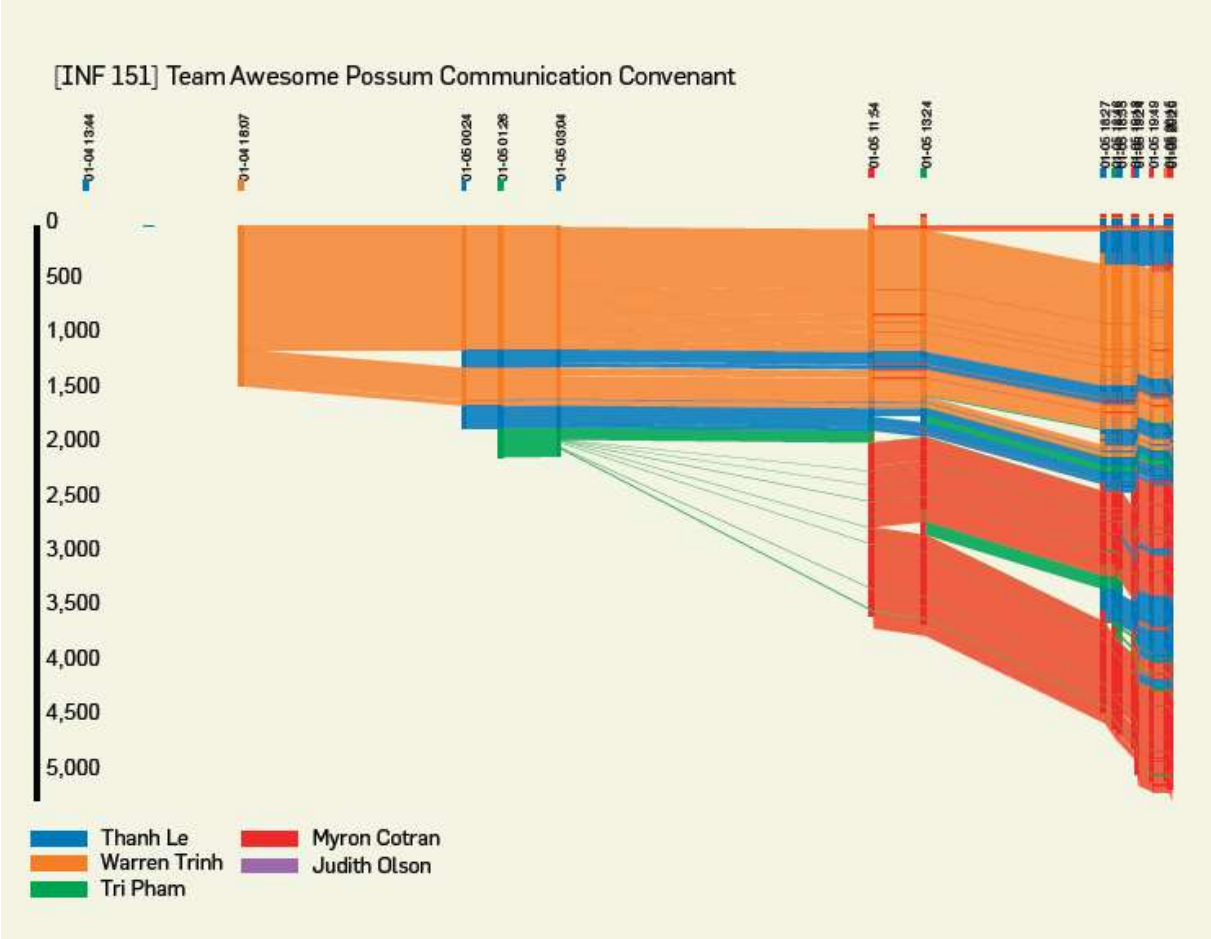
كان بين أيدينا وثيقةُ ShrEdit واحدة مفتوحة، مع مخطَّطٍ إجماليٍّ كنتُ كَتَبْتُه بنفسي، يُبرِز الموضوعاتِ التي قُدِّمت على مدى اليومين السابقين. بدأ الأشخاصُ الثمانيةُ جميعًا بتدوين ردودِ أفعالهم على الموضوعات وتقييمهم لها، وقرأ أعضاءُ اللجنة مُدخَلات (inputs) الآخرين وأضافوا إليها مدخلاتهم الشخصية. وكانت لهم الخِيَرَةُ في الكتابة حيثما شاؤوا؛ فكثيرًا ما كانوا يضيفون إلى كتابات غيرهم، وأحيانًا يعقدون جلساتِ مناقشةٍ اعتمادًا على النصوص.

بقي أعضاءُ اللجنة يعملون لأكثر من ساعةٍ على طباعة الوثيقة معًا وبصفةٍ متواقتة، وخرجوا بوثيقةٍ مؤلَّفةٍ من إحدى عشرة صفحةً، كانت غيرَ مرتَّبةٍ في شكلها، غير أنها غنيَّةٌ في مضمونها. وفي آخر الجلسة قال أحدُ أعضاء اللجنة متفكِّهًا: «أين زرُّ التنظيف؟»، انطلقت على إثرها من الجميع ضحكةٌ كانوا فعلًا بحاجةٍ إليها. ثم تطوَّع رئيسُ الفريق لإخراج نسخةٍ مبيَّضةٍ من هذه المسوَّدة، متَّخِذًا دَوْرًا شبيهًا جدًّا بدور ريكاردو في رواية Build-It. كان رئيسُ الفريق راضيًا تمامًا عن حجم المادّة الأوَّلية التي سيُبنى على أساسها؛ فقد كانت أغنى كثيرًا منها لو كانت نقاشًا جماعيًّا لوقائع جلسة. وعندما عادَ أعضاءُ اللجنة أدراجَهم كانوا قد أتمُّوا مهمَّتَهم جميعًا إلا رئيسُهم لمّا يُتِمَّها.

رؤًى في الرواية. اللافت في هذه الرواية هو أن الكتابة فيها لم تكن مسألةَ «فَرِّقْ تَسُدْ»، شأنَ الروايتين السابقتين، بل ما يمكن تشبيهُه بـ«خليَّةٍ نَشِطة» بسبب كثرة عدد الكُتّاب المنخرطين في هذه العملية من دون تنظيمٍ مُحكَم. يلاحَظ أن كلَّ مشارِكٍ أسهمَ في كلِّ مقطعٍ من العمل كما يشاء ويختار، فيَطبع أحيانًا على مقربةٍ دانيةٍ من المدخل الراهن لمشاركٍ آخر. وطوال ساعةٍ سادَ الغرفةَ صمتٌ وسكون، إلّا من ضحكةٍ تنطلق فينةً بعد فينة، ووَقْع نقراتٍ لطيفةٍ للمفاتيح على اللوحات.

الرواية 4: طلّابٌ يكتبون واجباتهم المدرسيَّة في غرفة الصف باستعمال تطبيق Docs من غوغل، ترويها جودي. اعتادَ طلّابي في مقرَّر إدارة المشروعات العملَ ضمن مجموعاتٍ لإنجاز مشروعٍ صغيرٍ على مدى ربع السنة الدراسية. وكان عليهم تقديم عدة وثائق كانت مألوفةً في العُرف الرسمي لإدارة المشروعات، من مثل: دراسة حالةٍ في مجال الأعمال، وتقديم بيانٍ استشرافي، وهكذا. وتعيَّن عليهم كذلك استعمال نظام Docs من غوغل والمشاركة بوثيقتهم النهائية معي شخصيًّا لتقييمها.

بموافقة الطلبة، قمنا بدراسةٍ تحليليةٍ لوثائقِ ثلاثِ سنوات – بلغت في مجموعها 96 وثيقة؛ فنظرنا في أساليب الكتابة التشاركية وأنماطِ العمل فيها، ورَبَطنا بعضَ السِّمات الأساسية بدرجة جودة الوثيقة النهائية. تجدر الإشارةُ هنا إلى أن أنماطَ العمل كُشِفَت بواسطة أداةٍ تسمّى DocuViz[11]، تَعْرِض صورةً لتأريخ المراجعة (revision history)، مع شرائح تأريخٍ زمنيةٍ تُعَرِّف بشخصيات الكتّاب وتفصِح عمّا كتبوا ومتى كتبوه. ويُظهِر الشكلُ المرافقُ عرضًا مرئيًّا للخط الزمني (timeline visualization)، كما أظهرته الأداة DocuViz لأسلوب مجموعةٍ واحدةٍ من المشاركين. يمثِّل الشكلُ الطلبةَ بألوانٍ عدَّة،



عرضٌ مرئي، كما أظهرته الأداة المسمّاة DocuViz، لفريقٍ في جلسة عملٍ متواقت [كتابة تشاركية] قُبَيْل نهايتها.

ويشير حجمُ الخط الملوَّن إلى مقدار الإسهام، في حين تشير الشرائحُ الزمنيةُ إلى تاريخ إنتاجه. وقد تبيَّنَ أن هذا الفريقَ كان عملُه المتواقتُ غزيرًا في المراحل الأخيرة من العمل.

وقد دَلَّلَ قرابةُ 95% من تلك الوثائق على وجود كتابةٍ متواقتة، بحسب تعريفنا لمصطلح «التواقُت» على أنه فعاليةُ كتابةٍ في غضون سبع دقائق من آخر عمل دون إغلاق الوثيقة وفتحها. ولعلَّ ثمة مَنْ يتوقَّع أن جلسات العمل المتواقت هذه تندرج تحت أسلوب «فَرِّق تَسُدْ». وإذ إن نحو الثلث منها كان فعلًا كذلك، فإن معظمها أَظهَرَ عمليةَ التحرير وقد أنجزَها عدةُ أشخاص في الفقرة نفسِها، أو حتى في خليّة الجدول ذاتِها.

رؤًى في الرواية. لم يتوفَّر لدينا في هذه الرواية من التفاصيل إلا القليل، لجهة الطريقة التي تَمَكَّنَ الطلبةُ فيها من تنجيز أعمالهم وتحريرها والتدقيق فيها. على أنَّ تتبُّعَ تأريخ المراجعاتِ لا يُظهِر إدارةَ مشروعاتٍ صريحةً تتمثَّل في ندب أشخاصٍ لكتابة فقراتٍ فحسب، بل أيضًا تحريرًا حرًّا لمَداخلهم الخاصة، وكذلك –وهذا مهمّ– مداخل الآخرين. لوحظَ أيضًا قدرٌ كبيرٌ من العمل المتواقت.

الرواية 5: عرضُ وقائع جلسةٍ وإنشاؤها تشاركيًّا باستعمال نظام ShrEdit، يرويها غاري. إضافةً إلى استعمال نظام ShrEdit لإدارة البحوث[8]، فقد استعملتُه بنفسي لتدوين محاضر اللقاءات البحثية. في هذه اللقاءات كانت وثيقةُ ShrEdit مشترَكةً ومفتوحةً على محطة عمل كلٍّ من المشاركين، وغالبًا ما كنّا نُسْقِطُها على شاشةٍ كبيرة في صدر الغرفة. وكان أحدُنا –أنا عادةً– يأخذ زمامَ القيادة، لكنَّ أيًّا منَّا كان يمكن أن يتولّى أمرَ التحرير.

يقوم المشاركون في الاجتماع عادةً بتصحيح الأغلاط الطباعية وغيرها من الأخطاء (كتصحيح تهجئة اسم شخصٍ ما) التي وقعتُ فيها أنا أو غيري من الكَتَبَة الناسخين أثناء الطباعة السريعة؛ فطالما كنّا نعلّق على مظهر الوثيقة قائلين إنها تبدو وكأنَّ باكمان (Pac-Man)(‡) يجري في أَثَرِ مدوّن الوقائع يستثيره ليتصيَّده. وقد ساعدَ المرأى المشترَكُ للملاحظات (سواء المرأى المُسْقَط على الشاشة أو آراء الأفراد العاملين على حواسيبهم) في الحفاظ على التركيز الذهني لأفراد المجموعة، والتوثُّق من أن النِّقاطَ قد التُقِطت على وجهٍ صحيح. ولا شكَّ في أنَّ هذه الأداة قد مكَّنتْ من تحقيق لقاءاتٍ فعّالة وملاحظاتٍ أعلى دقَّة.

يكون الكَتَبَةُ في مثل هذه اللقاءات عادةً منهمكين إلى درجةٍ لا يستطيعون معها الإسهامَ أو المشاركةَ بسهولةٍ في المحادثات والمناقشات. أما عند استعمالنا لنظام ShrEdit، فقد بات بإمكان شخصٍ آخرَ أن يضطلعَ بدور الكاتب في حين يتكلَّم الكاتبُ الرئيسيُّ ليكون مُسْهِمًا كاملَ المشاركة في الاجتماع.

في أحد اجتماعاتنا المهمَّة مع مؤسَّساتٍ راعية، أسقطنا وقائعَ الجلسة على شاشةِ عرضٍ في الغرفة يراها الجميع. وكان علينا أن نتناولَ بالبحث تسعةَ بنودٍ على جدول أعمال ذلك الاجتماع الذي استغرق ساعاتِ ما بعد الظُّهر كلَّها. وبعد عدة ساعاتٍ من الكلام وتدوين ملاحظاتٍ ضافية، خرجَ منسِّقُ الاجتماع لينبّه إلى أننا لم نستوعب في بحثنا سوى أوَّلِ بندَيْن على جدول الأعمال، وأنَّ سبعةً من البنود ما زالت تنتظر منّا البحثَ، فكان علينا أن نتابع. ثم بَرَزَ مشاركٌ من إحدى المؤسسات الراعية يقول إنه ليس معنيًّا بأيٍّ من البنود الباقية، لكن ما زال في جَعبته الكثيرُ ممّا يمكنه قوله في صُلب موضوع البحث المطروح. وبناءً على ذلك، تابعنا نحن بحثَ بنود برنامجنا، في حين انصرفَ هو يكتب في الجزء الذي يخصُّه، بعيدًا عن أعيننا. وقد عنونَ الجزءَ الخاصَّ به: «لم تطَّلِعوا على هذا بعدُ»، واستمرَّ في تدوين أفكاره ساعةً من الزمن كان بإمكانه أثناءها أن يعود إلى أجزاءٍ سابقةٍ من وقائع الجلسة يستعين بها، في الوقت الذي أتممنا نحن فيه بنودَ جدول أعمالنا عن آخرها. وهكذا حقَّقنا غايتَيْن بعملٍ واحد: فوَقْتُه كان مَصُونًا، ومهمَّتُنا منجَزَةً. ولمّا كانت هذه الوثيقةُ واحدةً (لا إضافةً لأفكاره من وثيقةٍ أخرى)، كان ذلك حَرِيًّا أن يجعل إضافاتِه أيسرَ قراءةً في سياقها.

رؤًى في الرواية. كان للوثيقة المشتركة هنا، كما في الروايات الأربع السابقة، شكلٌ مُعَدٌّ سَلَفًا (جدول الأعمال). وفي حين قَسَّمَتْ ثلاثٌ من الروايات السابقة العملَ فأسْنَدَتْ إلى المشاركين وظائفَ محدَّدةً لإنجازها، تولّى شخصٌ واحدٌ –في الرواية الخامسة هذه– وظيفةَ الكاتبِ الرئيسي. واستطاع المشاركون الآخرون –بصمتٍ وعلى التوازي– أن يضيفوا تفاصيل ويصحّحوا أخطاءً. واستطاع أحدُ المشاركين أن يتابع إسهامَه في الوقت الذي تابعَ فيه المؤتمِرون الآخرون بحثَ بنود برنامجٍ لم يكن هو منخرطًا فيها، وبذلك لم يكن ثمة وقتٌ ضائع.

(‡) شخصيةٌ خياليةٌ في لعبةٍ فيديويةٍ للأطفال واليافعين تحمل اسمَ بَطَلِها هذا. يجول باكمان في متاهةٍ، ملتهمًا كلَّ ما يصادفه في طريقه. [المترجم]

الرواية 6: إنشاء وقائع جلسةٍ تشاركيًّا ومن بُعد باستعمال نظام Docs من غوغل، يرويها غاري. تَعْقِد اللجنةُ التنفيذيةُ لمجموعة SIGCHI[§] التابعةُ لاتحاد الآلات الحاسوبية (Association for Computing Machinery, ACM) لقاءاتٍ دوريَّةً تدوم يومين أو ثلاثة، ويوزَّع جدولُ الأعمال سلفًا، ويجري تعديلُه عن طريق البريد الإلكتروني كلَّما أُضيفت بنودٌ جديدةٌ إليه. ونحن نستعمل تطبيقَ Docs من غوغل لتدوين الوقائع، التي تتمثَّل بالوثيقة الناشئة عن إدراج جدول الأعمال من البريد الإلكتروني. وغالبًا ما يُعاد تنظيم جدول الأعمال ويجري تعديلُه في أثناء العمل.

وكما في الرواية السابقة، يعيَّن أحدُ الأعضاء كاتبًا رئيسيًّا، لكنَّ أيًّا منّا يمكنه أن يحرِّر في أيّ وقت؛ وعندما يتكلَّم الكاتبُ يضطلع آخرُ بدوره مؤقَّتًا.

ويَحدث، من حينٍ إلى آخر، أن يَحْضُر بعضُ الأعضاء أجزاءً من الاجتماع، بل حتى الاجتماعَ كلَّه، من بُعْد. وكثيرًا ما ندعو الأعضاءَ المتابعين من بُعد للمشاركة في بنودٍ محدَّدةٍ على جدول الأعمال، مع أنهم يتابعون في الوقت نفسِه أجزاءً أخرى من الاجتماع. وكان بعضُهم ينضمُّ إلى الاجتماع *دون* أن يكون مزوَّدًا بأيّ وسيلة ربطٍ سمعيةٍ أو بصرية (audio/video connection)، وفي هذه الحالة يؤدّي نظامُ Docs وظيفةَ أداة ائتمار (conferencing tool) متواضعة. إنَّ الانخفاضَ الشديدَ لمعدَّل التغيُّر في الوثيقة ملائمٌ لأيّ مشارِكٍ بعيد، إذ بإمكانه عندئذٍ أن يبتدل المهامَّ (task switch) في الأوقات المناسبة. أَشَرْنا أيضًا إلى أن المتحدِّثين ممَّن ليست الإنكليزيةُ لغتَهم الأمَّ يستفيدون من مَحاضِر الجلسات لكونها شكلًا من «العروض النَّصِّيَّة المُغْلَقة» (closed captioning)[**].

وكثيرًا ما يَستعمل أحدُ أعضاء اللجنة –في عرض تقريره– شرائحَ PowerPoint، التي تُدرَج في العادة مباشرةً في وثيقة نظام غوغل المشتركة، مع أن ثمة أنواعًا أخرى من المواد تُدرَج في هذه الوثيقة في بعض الأحيان، من قبيل روابط لمواد وِبْ ذات صلةٍ بموضوع بحثنا.

وفي اجتماعٍ منتظم آخر لمجموعة البحث، لا تُحذَف وقائعُ الجلسة أبدًا، بل إنها تُدفَع (تزاح) إلى الأسفل، بحيث يَظهر الجديدُ من جدول الأعمال والوقائع في الأعلى. وقد أثبتَ هذا الأرشيف نجاعتَه في مواطنَ عديدة تمكِّن المرءَ أن «يبعث» بندًا منسيًّا في جدول.

رؤًى في الرواية. تشترك هذه الروايةُ مع الرواية الخامسة لجهة التشاركية في تدوين وقائع الجلسات، لكنها تضيف مزيَّةَ استعمال الوثيقة المستحدثة مثل: الائتمار بعرض حزمة منخفض (low bandwidth) والعروض النصِّيَّة المغلقة.

الرواية 7: التعليم من طريق ملاحظة الأشياء في عملٍ متواقتٍ باستعمال منتَج Aspects، ترويها جودي. كان من المقرَّر أن يقدِّم أحدُ طلّابي السابقين في الدراسات العليا –وقد تخرَّج إلى عهدٍ قريبٍ ويعمل في جامعةٍ أخرى– عرضًا يتناول إسهامَنا المشترك في مؤتمرٍ سيُعقَد وشيكًا. طلبتُ إليه أن يُعِدَّ عَرْضَه باستعمال أداة عرضٍ من Aspects، وأن يُشرِكني بها.

وفي جلسةٍ كنّا –كلانا– نستطيع معاينةَ العرض ونحن نتحدَّث على الهاتف، قدَّمَ طالبي السابق عرضَه بلا مقاطعة

[§] Special Interest Group for Computer-Human Interaction: مجموعة الاهتمامات الخاصة للتآثر الإنساني–الحاسوبي.

[**] خدمةٌ أو خاصِّيَّةٌ تحوِّل المحتوى السَّمعيَّ أو البصريَّ لبرنامجٍ تلفزيّ أو فيديويّ إلى نصٍّ مكتوبٍ على الشاشة، يَظهر عادةً عند تفعيله. ابتُدعت هذه الخدمة أصلًا لمساعدة ذوي الإعاقات السمعية، لكنها مفيدة أيضًا في حالاتٍ أخرى كثيرة. [المترجم]

لأغراضٍ توقيتية. وفي الختام قدَّمتُ أنا ملاحظات عامةً تتَّصل بالتوقيت وترتيب الأشياء، واقترحتُ إدراجَ أشكالٍ أخرى غير تلك التي اختارها المحاضِر. وبينما كان هو منشغِلًا بالبحث عن الأشكال الجديدة وإثباتها في أماكنها من العرض، أجريتُ بعضَ التغييراتِ الشكلية الصغيرة على عددٍ من الشرائح انطلاقًا من آخرها. وعندما عاد طالبي السابق وقد أتمَّ استبدالَ الأشكال، قال مُقِرًّا: «أوه، أنا أرى ما تفعلين.» ثم إنه قامَ بتغيير الشرائح السابقة لتنسجم مع أسلوب التغييرات التي أدخلتُها. وناقشنا بعضَ التغييرات الأخرى فيما يخصُّ اختيارَ الكلمات والعبارات وصوغَها. حتى إذا صرنا راضيَيْن بما توصَّلنا إليه من نتائج، تقدَّمَ المحاضرُ بالعرض مجدَّدًا بدون مقاطعة. وبعد إجراء تغييرَيْن طفيفَيْن آخرَيْن، انتهت مهمَّتُنا.

كان لغاري تجربةٌ مشابهةٌ في المشاركة باستعمال شاشة Centra، مع القدرة على الكتابة تزامنيًّا. فقام وفريقُه بتمحيص مسوَّدة عرضٍ ومناقشتِها وإدخال تعديلاتٍ عليها بالزمن الحقيقي؛ فكان المشهدُ المشتركُ يحدِّد وجهةَ النقاش، والتحريرُ المتواقت يسرِّع عمليات المراجعة.

رؤًى في الرواية. إن ما كان يمكن أن يستغرق في العادة ساعاتٍ أو أيامًا من الأخذ والرَّد استغرقَ بالفعل ساعةً واحدةً ليس غير. فنمذجة أنواع التغييرات والتعديلات التي أردتُها أنا أتاحت لطالبي السابق أن يتقبَّلها دون أدنى نقاش. وكان من الوارد أن يثير اعتراضاتٍ وجدالًا، غير أنه في هذه الحالة لم يفعل أكثر من أن نَسَخَ تعديلاتي في الأسلوب كما هي دون اعتراضٍ ولا مناقشة.

الرواية 8: كتابة هذه الوثيقة [المقال الذي بين أيدينا] باستعمال نظام Docs من غوغل: كيف نفَّذنا ذلك. أمّا وقد كَشَفْنا –نحن الأربعة– عن رواياتنا في الكتابة المتواقتة، فقد اقترحَ ريكاردو أن نؤلِّفَ مقالًا في هذا الموضوع باستعمال طريقة Build-It التوثيقية. وافقنا جميعًا، إذ رأينا في ذلك تحقيقًا لفائدةٍ مزدوجة: كتابةِ هذا المقال، واختبارِ طريقة Build-It. واستنادًا إلى ذلك حدَّدنا لأنفسنا يومًا للكتابة بعد نحو شهرٍ من لقائنا المبدئي هذا، وذلك باجتماع ثلاثةٍ منّا شخصيًّا وانضمام دان لنا من بُعْد.

وتمهيدًا لهذا اليوم عَقَدْنا، قَبْلًا، مؤتمرًا فيديويًّا تدارسنا فيه الإطارَ العامَّ للمقال العتيد، وأنواعَ الروايات التي يمكن أن نستعملها. كلُّ ذلك النُّقِطَ في وثيقةٍ اشتركنا في كتابتها باستعمال طريقة غوغل. وأنشأ ريكاردو مخطَّطًا إجماليًّا مؤقَّتًا للعمل اعتمادًا على هذه المناقشة. وعندما اجتمعنا فعلًا، بحثنا في الساعة الأولى أفكارًا وتوجُّهات. لكنَّ ما ثبَّطَنا في البداية أنه لم يكن في أذهاننا بعدُ فكرةٌ واضحةٌ عن موضوع المقال؛ فطريقة Build-It مُعَدَّةٌ لاستراتيجية «فَرِّقْ تَسُدْ»، لا للمرحلة التي يتعيَّن علينا فيها الكشفُ عمّا نريد قولَه. أما المخطَّط الذي أنشأه ريكاردو فقد وجدناه قاصرًا عن استيعاب جميع جوانب المناقشة التي كنّا بصددها، فتخلَّيْنا عنه.

في نهاية المطاف، استقرَّ رأيُنا على أنَّ أفضل الخِيارات للخطوة التالية إنما تتمثَّل ببساطةٍ في أن ندوِّن مجموعةَ الروايات التي هي بمنزلة المادَّة الأوَّلية لعملنا. فكان بإمكاننا أن نقرأ في الوقت الذي يكتب فيه الآخرون. وكثيرًا ما كنّا نقرأ شيئًا كان غيرُنا قد كَتَبَه، ثم يعود كلٌّ منّا ليضيفَ أو يغيِّر أو يعدِّلَ في إسهامه الخاص. حتى إذا فَرَغنا جميعًا من ذلك قرأنا كلَّ ما كتبناه في سياقٍ أفضى بنا بعدئذٍ إلى مناقشة تنظيمٍ أفضلَ للمقال المأمول.

وواقع الأمر أننا، في كتابة التفاصيل ثم قراءة رواية كلٍّ منّا للآخر، وقفنا على بعض أوجه التشابه والاختلاف، فاستحدثْنا جدولَ عملٍ (working table) في الوثيقة مباشرةً لبيان جوانب التشابه وجوانب الاختلاف في الروايات، آملين أن

يكون هذا عونًا لنا في ترتيب الروايات ترتيبًا منطقيًّا، ومن ثَمَّ دعم تطوير الجزء الخاصّ بالمناقشة.

وفي غضون إنشاء هذا الجدول كان كلٌّ منّا يدرك أنَّ ثمة شيئًا ما أسقطناه من رواياتنا. وما إن وقفنا على أوجه التشابه الناشئة هذه حتى رحنا نستقري ذاكرتَنا ننقِّب فيها عن مزيدٍ من الأمثلة، فأطلقنا على هذه النقاط الإضافية التي جمعناها اسمَ «الظواهر الملحَقَة» (epiphenomena).

بعد قرابة ستِّ ساعاتٍ من المحادثات والكتابة، ثم المحادثات ثانيةً، شعرنا بالإرهاق، ومع ذلك كان علينا أن نقرِّر رسم خطواتنا التالية، وكان على أحدنا أن يقوم بالنقلة الأولى لترتيبِ الروايات وجعلِها متجانسةً ما أمكن من حيث الأسلوب، ثم كتابةِ مسوَّدةٍ للجزء الخاصّ بالمناقشة، باتِّخاذ جدول العمل إمامًا. وقد تطوَّعَتْ جودي للنهوض بهذه المهمَّة.

اقتضت المرحلةُ التاليةُ هذه من أحدنا أن «يحوز» الوثيقةَ مؤقَّتًا، دون أن يقوم الآخرون بعملية التحرير، ذلك لأن إعادةَ تنظيم المناقشة وإعدادَ مسوَّدتها الأولى تطلَّبَ الإحاطةَ بالوثيقة كاملةً بغية إنشاء الارتباطات المناسبة. وعندما انتهت جودي أشعرت الآخرين. على أننا لم نكن راغبين في أن يقوم الجميعُ بالتحرير معًا وفي وقتٍ واحد. ووَقَعَ اختيارُنا باديَ الرأي على ريكاردو –وهو كاتبٌ تقنيّ– للقيام بعملية نَسْخٍ وتحريرٍ أكثر تفصيلًا، فبَذَلَ وُكْدَه لموالفة الصوت مع سرعة الكتابة، وتحديد سمات الوثيقة التي يبدو أنها تُفسِد انسيابيةَ العمل.

ثم إننا عقدنا مؤتمرًا فيديويًّا تناولنا فيه عدةَ مسائل، ونظَّمنا لائحةً بالأعمال التي يتعيَّن علينا إنجازُها. وطلبنا إلى جودي وغاري أن ينفِّذا النقلةَ الأولى على الأجزاء التي تخصُّهما من اللائحة. ثم قام ريكاردو ودان بمزيدٍ من التحرير أو التعليق. نعود لنؤكِّد مدى أهمية الشخص الذي كان في موقع القيادة والتحكُّم.

على أننا لم نتطرَّق إلى الطريقة التي سنُدير بها التحريرات، وهل سنكتفي بإمضاء التغييرات والاعتماد على تأريخ المراجعة لتسوية النقاط التي تباينت فيها آراؤنا، وهل نستعمل نمط «اقترح» لنظام Docs الذي يشبه نظام وورد (Word) في تتبُّع التغييرات، أم نستعين بالتعليقات لمناقشة التغييرات وإقرارها؟ أما جودي وريكاردو فأجريا التغييرات مباشرةً، وأما دان فقد عَمَدَ إلى التعليقات ليقول مثلًا: «هذه النقطة تحتاج إلى إيضاح.» وبَيَّنَ دان لاحقًا أنه لا يريد تحريرَ عمل شخصٍ آخر. تشير هذه الأمثلة إلى أنَّ الناسَ يُقْبِلون على الوثيقة بأفكارٍ مختلفةٍ تتعلق بتحديد مَنْ هو المخوَّل أو المسؤول عن تغيير الأشياء. وفي هذا المساق، خَلَصَ بيرنهولتس وإيبارا[1] إلى أن إدخالَ تغييراتٍ على كتابات الآخرين تصرُّفٌ اجتماعيٌّ له تبِعاتٌ تتَّصل بالثقة والعلاقات.

واستلزَمتِ المرحلةُ الأخيرةُ من هذا العمل التشارُكي تضمين وثيقة نظام غوغل في برنامج Word ليلائم التنسيق المطلوب المكوَّن من عمودَيْن، المعتمَد في هذه المجلَّة الدَّورية. وبدءًا من تلك النقطة أصبحت عمليةُ التحرير تناقليةً (hand-off) بالكلِّيَّة، مع بيانٍ واضحٍ للمسؤوليات وخطوط الزمن.

رؤًى في الرواية. خلافًا لطريقة Build-It، فقد انبثقَ مخطَّطُ هذا المقال بعد كتابةِ الروايات، وتَشارُكِها، ومناقشتِها. واقتضى هذا المقالُ أيضًا استعمالَ الوثيقة بوصفها «مكان مَسْك» (holding place)، للاحتفاظ بالأشياء (كالجدول) التي قد تساعد في الكتابة، أو بالمادَّة التي تُحْذَف في نهاية المطاف. كان ثَمَّةَ أثناءَ عملنا أوقاتٌ كانت فيها الكتابةُ المتواقتةُ بأسلوب «فرِّقْ تَسُدْ» ملائمةً؛ وحالاتٌ تعيَّنَ على شخصٍ واحدٍ منّا (جودي) أن يتولى زمامَ الأمر وفي الوقت نفسه أن يلتقط تنظيمَ المقال البازغ؛ ومَواطنُ كان فيها التناقُلُ المتسلسل (serial handoff) للتحرير هو الحلَّ المناسب. وتنبَّهنا إلى أنه كان علينا،

لجهة التغييرات، أن نقرّر صراحةً: هل نكتفي بالتحرير أم بالاقتراح أم بالتعليق؟

المناقشة

إن القدرةَ على الكتابة المتواقتة في وثيقةٍ مشتركةٍ لهيَ مظهرٌ رصيفٌ للتقدُّم التقاني. بيد أن الأدبيّات (literature) التي تناولت هذا الموضوع لم تَذْكُرْ إلا اليسيرَ عن العملية الاجتماعية التي تسخِّر التقدُّمَ التقانيَّ في الكتابة المتواقتة تسخيرًا حقيقيًّا لمصلحة المستعملين. وتحاول هذه الرواياتُ أن تلقيَ الضوءَ على هذه العملية الاجتماعية بالذات.

في هذه الروايات جُلِّها كان ثَمَّ شخصٌ يقود العملَ بإنشاء نوعٍ من البِنية: كالشجرةِ في نظام IDE، والمخطَّطِ الإجماليّ في نظامَيْ ShrEdit وDocs، وجدولِ الأعمال في الاجتماعات، ومسوَّدةِ العُروض في نظامَيْ Aspects وCentra. وكان الاستثناءَ الوحيدَ كتابةُ هذا المقال. عقدنا جلسةَ النِّقاش بعد أن كتبنا رواياتِنا وقرأ كلٌّ منّا روايةَ الآخر، فانبثقت البِنية.

وللكتابة المتواقتة منافعُ عديدةٌ منها: مكاسبُ في الإنتاجية، وشعورٌ عميقٌ بالرضا عن وقتٍ أُحسِنَ استثمارُه، ودُرْبَةٌ عمليةٌ بمحاكاة أسلوب المشارِك. وعلى مستوًى تكتيكي، يستطيع المشاركون الانتقالَ بسرعةٍ نحو وثيقةٍ عالية الجودة لأن بإمكانهم معاينةَ ما يفعله الآخرون ومضاهأتَهم (ومحاكاتهم) فيه. وبانضمام الأشخاص إلى الكتابة يستطيعون الاطِّلاعَ على أحدث الأعمال بقصد جَعْل إسهاماتهم منسجمةً مع الرؤية الإجمالية. إنَّ للقدرة على العمل تزامنيًّا على وقائع الجلسات فوائدَ أبعد من مجرَّد تسجيل المحتوى وتصحيحِه؛ فالجميع كانوا مجتمعين «على الصفحة نفسِها.»

هذا لا يعني، بطبيعة الحال، أن جميعَ الفعاليات التشاركية يمكن أن تفيد من العمل المتواقت؛ فهناك حساسيّاتٌ تعتمل في نفسك عندما يقوم شخصٌ بتغيير كتابتك، وحساسيّاتٌ قد تُستثار عندما تعلم أن آخرين يمكن أن يطَّلعوا على سيرورة كتابتك (مثلًا: إذا كنتَ بطيئًا في الكتابة أو ضعيفًا في تهجئة الكلمات). وقد يرى البعضُ في معاينة تحريرات الآخرين وهم يكتبون أو يحرِّرون أمرًا صارفًا للانتباه والتركيز. ثم إنَّ العملَ التزامنيَّ غيرُ مُجْدٍ أبدًا إذا انعدمت الثقةُ بينك وبين زملائك، ربما بسبب الجوِّ التنافسي السائد، أو لأن خبرتَهم وقدراتِهم لا تكافئ خبرتَك وقدراتِك. بل قد يصل الأمرُ، في بعض الثقافات التنافسية، إلى حدوثِ تخريب. يُضاف إلى ذلك أن كثيرًا من جلسات العمل التزامني التي وَضَعْنا مخطَّطَها الإجماليَّ كان يجري على المسوَّدات، لا على النسخة النهائية (مع أن الرواية 7 تحتوي على إعادة التنسيق التشاركي للعرض النهائي).

كذلك قد تنشأ مشكلاتٌ تقنية كحروب التحرير (edit wars)، ذلك لأن المرءَ في التحرير المباشر لا يرى الأصلَ الذي حصل منه التغيير. وأيضًا، لو اتفقَ أن شخصَيْن كانا في النقطة نفسِها تمامًا، أحدهما يضيف والآخر يحذف، فإن من شأن ذلك أن يكون مُربِكًا جدًّا. ينضاف إلى ذلك أن الكتابة المتواقتة تتطلَّب من المرء أن يكون على شبكة، معتمدًا على ترابطيَّة المخدِّمات. وقد يسبِّب استعمالُ خدمةٍ في السحابة عند البعضِ مخاوفَ ذاتَ صلةٍ بالخصوصية (privacy).

أنماط استعمال الكتابة المتواقتة. ممّا يلفت النظرَ عندما نتفحَّص الرواياتِ أنْ ليس ثمة تنوُّعٌ كبيرٌ في التجارب، بل أنماطٌ قليلةٌ مختلفةٌ من الكتابة الجماعية. وتندرج الرواياتُ في مجموعتين اثنتين: أربعةِ أنماطٍ من العمل المتزامن، ونمطَيْن من عمليات التناقُل اللامتزامن (asynchronous hand-offs) المصاحِبة.

اعتَمَدَتِ الرواياتُ الأربعُ الأولى وتقريرُ كتابة هذا المقال استراتيجيةً من *فَرَقٍ تَسُدّ* متواقتة. وفي بعض مراحل إنشاء الوثيقة كان المؤلِّفون يكتبون في وقتٍ واحد، إلا أنهم كانوا في الأغلب الأعمّ يكتبون في فقراتٍ مختلفةٍ من الوثيقة.

تَستعمِل الروايتان 5 و6 كاتبًا رئيسيًّا، مع وجود كاتبٍ ثانٍ أو ثالثٍ إمّا ليضطلعَ بالمهمَّة فورًا عندما يكون على الكاتب الرئيسيّ أن يتكلم، وإمّا ليقوم بإجراء إضافاتٍ أو تحريراتٍ ثانوية. نسمّي هذا الإجراء نمط *الكاتب الدوّار* (rotating scribe).

وتَستعمِل الروايةُ 5 نمطَ تفريع (branching pattern)، وهذا يعني أنه إذا لم يكن أحدُ المشاركين منخرطًا في المحادثة المباشِرة، استُثمِر الوقتُ استثمارًا إنتاجيًّا لإجراء مزيدٍ من الكتابة يقرؤها الآخرون لاحقًا. فمن حيث الجوهر، يكون هذا الشخصُ منخرطًا في إنشاء فرعٍ (branch) جديدٍ في وقائع الجلسات، على حين يواصل الآخرون مهامَّهم. وهذا النمطُ هو شكلٌ من أشكال «فرِّقْ تَسُدْ».

أما النمطُ الرابع فمثالُه الرواية 3، ما نسمّيه *السِّرْب* (swarm)، وفيه يكون الجميعُ في الوثيقة، كلٌّ يكتب الجزءَ الخاصَّ به، ويقرأ ما كَتَبَه الآخرون، معلِّقًا أو مصحِّحًا. وليس ثمة مسؤوليةٌ فرديةٌ عن جزءٍ معيَّن؛ فالجميع مسؤولون عن الوثيقة برمَّتها. وهذا شبيهٌ أيضًا بالروايات التعليمية، حيث يمثِّل «المعلِّم» ما ينبغي أن يحاكيه «الطالبُ»، ثم يُتِمّان معًا الوثيقةَ بجهدٍ مشترك.

أما النمطُ الخامس، وهو *التهذيب* (cleanup)، فهو عملٌ منفردٌ يتبدّى في عدة روايات. على سبيل المثال، في الكتابة الخاصة بنا نحن، عقدنا جلسةً كبيرةً للكتابة المتواقتة، ثم تولَّتْ جودي وحدَها إعادةَ تنظيم الرواياتِ إلى مجموعات، وكتابةَ النَّص الذي يمثِّل النِّقاشَ المبيَّن في الجدول. وفي تقرير اللجنة، أَخَذَ رئيسُها على عاتقه مهمَّةَ تبييض النسخة المسوَّدة من النَّص وتركيبِ أجزاء المحتوى. وفي طريقة Build-It حَمَلَ ريكاردو مسؤوليةَ تهذيب النَّص وتنسيقِ الأسلوب، محوِّلًا النَّصَّ المسوَّدَ إلى نثرٍ صالح.

وأما النمطُ السادس، الذي كثيرًا ما تمثِّله أيضًا رواياتٌ عدَّة، فهو عملية *التناقُل* (hand-off)، حيث يتولّى عدَّةُ كُتّابٍ المسؤوليةَ مُدَدًا يعيدون فيها التنظيمَ ويستقصون الحقائقَ ويُجْرون تحريراتٍ بسيطة. وغالبًا ما كان هذا الإجراءُ مصاحِبًا جلساتِ العمل المتواقت.

نلحظ أن اجتماع المشاركين بالحضور الشخصي في الكتابة المتواقتة كان مهمًّا، لكنه لم يكن حتمًا؛ فهو يتيح نفاذًا مباشِرًا للمشاركين الآخرين لأغراضٍ من قبيل التوضيح، ومعاينة تعبيرات الأشخاص، والإيحاء بطلب دَوْرٍ للتكلُّم، وهكذا .. ومع ذلك، فإن العديد من الروايات حقَّقَتْ إسهاماتٍ ناجحةً من مشاركين بعيدين، حتى من دون ترابطيةٍ مسموعةٍ أو مرئية (audio or video connectivity). وفي إحدى الحالات لم يَرَ المشاركون من بُعد إلا وثيقة نظام غوغل المطوَّرة، علمًا بأن هذا المستوى من المشاركة كان كافيًا تمامًا لبعض الأغراض.

ظواهر ملحَقة. في أواخر نقاشنا لأوجه التشابه وأوجه الاختلاف بين الروايات، تعرَّضْنا لبعض الظواهر الملحَقة– المتمثِّلةِ في سلوكٍ غير اعتياديّ يتولَّد من واقع كون العمل ينشأ تزامنيًّا على مرأًى من الجميع.

فمن الظواهر الملحَقة ظاهرةٌ تضمَّنت حسَّ الدُّعابة، وهو مكوِّنٌ اجتماعيٌّ مهمٌّ للعمل المكثَّف. فمثلًا، في أثناء اجتماعاتٍ عُرِضَتْ فيها وقائعُ الجلسات على الشاشة، طالما كان غاري يدوِّن –بوصفه هو الكاتب– تعليقاتٍ فيها شيءٌ من التعريض بما صَدَرَ عن أحد المشاركين، ثم يمحو ما دوَّنه بسرعة. فعلى سبيل المثال، لو استرسلَ هذا المشاركُ في الكلام أو استطردَ، لكَتَبَ غاري، وبحروفٍ كبيرةٍ لا يطَّلع عليها هذا المتحدِّث، عبارة: «أما حانَ وقتُ الغداء؟» لكنه سرعان ما يحذفها!

قام طالبٌ من فريق إدارة المشروعات، وكان مستغرقًا في مناقشة مسألة التحرير المتواقت في جلسةٍ مطوَّلةٍ تتعلق بزبونٍ لهم، بإدراج صورةٍ هزليةٍ لرَجُلٍ أشيب يرتدي قَلَنْسُوَةً من فرو الدُّبِّ، تحمل العبارة: «هذا الفتى!» بقيت الصورةُ في الوثيقة قرابة دقيقةٍ ثم حُذِفَت. وقد قرأ أعضاءُ اللجنةِ الزائرةِ المناقشاتِ المنبثقةَ من داخل التقرير، فكانوا يُطْلِقون قهقهاتٍ من الضحك بين الحين والآخر. وأخيرًا عَلَّقَ أحدُ أعضاء اللجنة مازحًا: «أين زرُّ الحذف؟» انطلقت بعدها ضحكةُ ارتياحٍ توحي بإدراك المقصود، وتؤْذِن بانتهاء الجلسة.

ولاحظنا ظاهرةً ملحَقةً أخرى مفادُها أن الوضوح (visibility) عاملٌ حافز؛ فوضوح العرض المقدَّم عن مدى تقدُّم العمل في وضع كتابٍ مدرسيّ باستعمال نظام IDE كان في حدِّ ذاته دافعًا للناس. وبالمثل، لمّا كانت الوثيقةُ حيَّةً (live) ويُشتغَل عليها تزامنيًّا، فبإمكان المرء أن يرى أين وصلت الفعالية، وأن يكون هذا باعثًا له على القراءة باهتمام، ومناقشةِ أيّ قضايا تنشأ إمّا عن طريق المحادثة النَّصِّيَّة وإمّا بالتخاطُب الصوتي. ويلاحَظ أنه بوجود فعّاليةٍ واضحةٍ يشعر المشاركون أنهم ملزَمون بالتركيز على الوثيقة التي هي في قيد الإنشاء، وكانت إحداها تهتم بـ«النشاط الزلزالي». ونعتقد أن الطالبَ الذي أدرجَ صورةً هزليةً أثناء جلسة عملٍ متواقت قد استعان بروح الفكاهة لتحريك الباعث على استمرار العمل. وفي فريق عملٍ آخر، أدرجَ طالبٌ صورةً تمثِّل رفشًا، كَتَبَ فيها: «اعملْ بجِدّ!»

النتيجة

إنَّ الكتابةَ المتواقتةَ قدرةٌ فعّالةٌ جدًّا تتوفَّر اليومَ في البرمجيات التجارية على نطاقٍ واسع. وغالبًا ما تتمازج بنجاح مع شيءٍ من الكتابة التناقُلية والجلسات التي يتولى فيها شخصٌ واحدٌ زمامَ الأمور لدمج المادّة بالصوت. على أن التقانةَ وحدَها لا تصنع إنتاجيَّةً غنيَّةً رصينة، أما الإنسانُ فقادرٌ على ذلك. وما نعرِضُه هنا هو عددٌ من الروايات والتعليقات والشروح تتَّصل بما يمكن أن يجعل هذا اللَّونَ من الكتابة قوةً حقيقيةً فاعلة.

ونحن نُلحِق الرواياتِ بستَّة أنماطٍ من الكتابة كلَّما كان العملُ المتواقتُ ممكنًا. ويحتاج أعضاءُ فريق العمل الآن إلى التخطيط لأسلوب العمل (وبعضُه يتضمَّن كتابةً متواقتة) الذي ينسجم وطبيعةَ الأغراض القريبة؛ فإن بعضَ الأعمال التشاركية الإجمالية، كما يراها غلوشكو (Glushko)[3]، يجري التخطيطُ لها سلفًا (ما يسمّيه تشاركًا تراتبيًّا (hierarchical collaboration))، أو يطوِّرها المؤلِّفون بالاتفاق المشترك (ما يسمّيه تشاركًا إجماعيًّا (consensus collaboration))، أو تتَّخذ طابعَ المناقشة الحرَّة المفتوحة (ما يسمّيه تشاركًا مفتوحًا (open collaboration)). ولكلٍّ من هذه المقاربات، بطبيعة الحال، مَواطنُ قوةٍ وضعف؛ ومن هنا تبرز أهميةُ الاختيار الموفَّق للمهمة التالية. وقد تقدَّمَ لنا غيرَ بعيدٍ أن المنفعة لا تكون عامَّةً لجميع الأفرقاء وفي جميع الأحوال، فالتعاون والثقة هنا أمران أساسيّان.

وقد سُقْنا مجموعةً من الأمثلة اعتمدنا فيها طرائقَ جديدةً في العمل كانت مثمرةً جدًّا، ومن ثَمَّ أورثَتْ رضًا وقبولًا. وخطوتُك التالية هي أن تبادر إلى تحليلِ حالات الكتابة التشاركية التي تعيشها يومًا بيوم، وصَوْغِ أكثر الطرائق ملاءمةً، مع العلم بأن الاحتمالاتِ والإمكاناتِ غنيةٌ جدًّا.

كلمة شكر. تدين أجزاءٌ من هذا العمل بالفضل للدعم الماليّ الذي قدَّمته مؤسسةُ العلوم الوطنيةُ بموجب المنحة ذات الرقم ACI-1322304، وكذلك لجائزة البحث المركَّز من غوغل، الممنوحةِ لجودي وغاري أولسن [المشاركَيْن في تأليف هذا المقال]. والشكر موصولٌ لكلٍّ من: Tom Boellstorf وBonnie Nardi وBob Glushko وغيرهم من المراجعين والمدقِّقين

الذين لم تُذكَر أسماؤهم، على ما قدَّموه من أفكارٍ خلّاقةٍ وتعليقاتٍ مفيدةٍ على المسوَّدة الأولى للعمل.

المَراجع

[1] Birnholtz, J. and Ibara, S. Tracking changes in collaborative writing: Edits, visibility, and group maintenance. In *Proceedings of CSCW'12*, 2012, 809–818.

[2] Boellstorff, T., Nardi, B., Pearce, C., and Taylor, T.L. Words with friends: Writing Collaboratively Online. *Interactions 20*, 5 (Sept.-Oct. 2013), 58-61.

[3] Glushko, R. Collaborative authoring, evolution, and personalization for a 'Transdisciplinary' textbook. In *Proceedings of OpenSim 2015*.

[4] Group Technologies. Aspects: The first simultaneous conference software for the Macintosh, Version 1. Manual, (1991). Group Technologies, Inc., Arlington, VA.

[5] Halasz, F.G., Moran, T.P., and Trigg, R.H. Notecards in a nutshell. In *Proceedings of the SIGCHI-GI Conference on Human Factors in Computing Systems and Graphics Interface* (1987), 45–52.

[6] Irish, P.M., and Trigg, R.H. Supporting collaboration in hypermedia: Issues and experiences. *J. American Society for Information Science 40*, 3 (1989), 192–199.

[7] McGuffin, L. and Olson, G.M. ShrEdit: A shared electronic workspace. CSMIL Tech. Report, 1992. The University of Michigan, Ann Arbor, MI.

[8] Olson, J.S., Olson, G.M, Storrosten, M. and Carter, M. Groupwork Close Up: A Comparison of the Group Design Process With and Without a Simple Group Editor. *ACM Trans. on Information Systems 11*, 4 (1993), 321–348.

[9] Olson, J.S., Wang, D., Zhang, J. and Olson, G.M. How people write together now. *Trans. Computer-Human Interaction 24*, 1 (2017), 1–40.

[10] Russell, D.M., Burton, R.R., Jordan, D.S., Jensen, A-M., Rogers, R.A., and Cohen, J.R. Creating instruction with IDE: Tools for instructional designers. *Intelligent Tutoring Media 1*,1 (2009), 3–16.

[11] Wang, D., Olson, J.S., Zhang, J., Nguyen, T. and Olson, G.M. DocuViz: Visualizing Collaborative Writing. In *Proceedings of the Conference on Human Factors in Computing Systems*, (2015), 1865–1874.

المؤلِّفون

ريكاردو أولينيوا (Ricardo Olenewa): كاتبٌ تقنيٌّ يقيم في ووترلو، أونتاريو، كندا.

غاري أولْسن (Gary M. Olson) (gary.olson@uci.edu): أستاذٌ فخريٌّ في المعلوماتية بجامعة كاليفورنيا، إيرڤين.

جوديث أولْسن (Judith S. Olson) (jolson@uci.edu): أستاذةٌ فخريَّةٌ في المعلوماتية بجامعة كاليفورنيا، إيرڤين.

دانيال راسِلْ (Daniel M. Russell) (drussell@google.com): عالِمٌ باحثٌ متقدِّمٌ يعمل في Google، ماونتين ڤيو، كاليفورنيا.

كل شيء عن الصور

IT'S ALL ABOUT IMAGE*

Samuel Greengard
ترجمة: د. أميمة الدكاك
مراجعة: د. أحمد الحصري

تتطور تقانات تعرّف الصور بسرعة. يكتشف الباحثون طرائق جديدة لمعالجة المهمة دون الحاجة إلى مجموعات معطيات هائلة.

ليست مهمة اكتشاف الكون منوطة بالخجولين وقليلي الصبر؛ ثمة حاجة لإمعان النظر في أعمق ما وصلنا إليه فيما يتعلق بالفضاء الخارجي، ومحاولة إدراك فحوى المجرات البعيدة، والنجوم، والسحب الغازية والكوازارات[1](quasars) والهالات والثقوب السوداء. يقول كيفن شافنسكي (Kevin Schawinski) عالم الفلك والأستاذ المساعد في معهد الفضاء في المعهد الفيدرالي للتقانات (ETH) في زيوريخ "إن فهم سلوك هذه الأشياء، وكيفية تفاعلها فيما بينها يعطينا إجابات عن كيفية تشكّل الكون وكيفية عمله".

تكمن المشكلة في أن الأجهزة التقليدية كالمقاريب (telescopes)، حتى عند اشتمالها على تطورات جذرية في البصريات ووضع مراصد منها في الفضاء، حيث يغيب غبار الأرض وضوؤها، لا يمكنها أن ترى إلاّ ضمن مدىً محدود. فعلى سبيل المثال، لقد غيّر منظار هَبِل (Hubble) الطريقة التي يرى فيها علماء الفضاء والفلك الفضاءَ العميق، فزوّدنا بصورٍ أشدّ وضوحًا بكثير مما كان ممكنا في السابق. وبالطبع فإن هذه الصور ترتبط بالزمان والمكان ارتباطًا غير قابل للانفصام. يقول شافنسكي "إلا أنّ الصور لا تسمح لنا برؤية ما حدث في الماضي بقدر ما نريد، فكلما نظرنا أبعد في الكون أمكننا فهم أصل تكوّنه أكثر وإدراك كيفية تطوره".

لقد غيّر المعادلةَ الآن تدخّل واجتماع عالم تعرّف الصور حاسوبيا، والشبكات العصبونية الصنعية وعلوم المعطيات. وسمح لنا تدفّق حجوم هائلة من المعطيات، بإيجاد أجوبة على أسئلة تعذّر فهمها من قبل. فقد شرع العلماء، في السنوات الأخيرة، بتدريب الشبكات العصبونية على تحليل المعطيات القادمة من الكمرات في المقاريب المشيدة على سطح الأرض وفي الفضاء. وفي كثير من الحالات، تمكنت خوارزميات التعلم الآلي من تحسين الصور الضبابية وتحديد هوية الأجسام البعيدة على نحوٍ أفضل من تلك التي توصل إليها الإنسان.

**نُشِر هذا البحث في مجلة Communications of the ACM، المجلد 60، العدد 9، أيلول (سبتمبر) 2017، الصفحات 13 – 15.*

[1] جسم سماوي كبير وبعيد، يصدر كميات استثنائية من الطاقة، يبدو شبيها بالنجوم حين نراه بالمقراب (telescope). يتوقع احتواؤه على ثقوب سوداء ضخمة وقد يكوّن مرحلة من مراحل تطور بعض المجرات. (المترجم)

ويقول الدكتور فرانسوا لانوس (Francois Lanusse) الباحث في مركز ماك وليامز (McWilliams) للكونيات في جامعة كارنجي ميلون (Carnegie Mellon): "تحدث علوم المعطيات والمعطيات الكبيرة ثورةً في مجالات كثيرة من علوم الفضاء".

إن ضم المزيد من المعطيات إلى التطورات في علوم المعطيات وإلى الطرائق الجديدة التي تسمح للباحثين بتدريب شبكات عصبونية بسهولة وتكلفة قليلة، شجع العلماء عمليًا على أن ينظروا في مجالات لم يكونوا يرونها من قبل. ليس أقلها أهمية تلك التطورات التي لا تقتصر على علوم الفضاء والفلك؛ وإنما تعدّتها إلى صفيفة من مجالات أخرى، كتطوير العربات الذاتية الحركة والربوطات والطائرات المسيرة والهواتف الذكية وغيرها. كما أنها تُستَعمَل أيضا لفهم كل شيء فهمًا أفضل: بدءًا من معرفة كيفية إسهام النماذج اللغوية في نشوء النَّزْعة العنصرية، إلى تحديد الشدة الممكنة للأعاصير حين تشكلّها.

يقول جيف كلون (Jeff Clune) الأستاذ المساعد في علوم الحاسوب في جامعة يومنغ (Wyoming): "حتى وقت قريب جدًّا، لم تكن الحواسيب ترى العالم أو تفهمه جيدًا. ولقد أحدث إمكان تدريب الشبكات العصبونية بسهولة وبسرعة تحويلًا عميقًا في تعرُّف الصور ومكّن من إحراز اختراقات مذهلة".

اكتمال الصورة

ليست الشبكات العصبونية الصنعية جديدة، إذ تعود فكرتها إلى أربعينيات القرن الماضي، وقد جربها الباحثون على مدى ربع القرن الماضي. ومع ذلك، فقد نضجت التقانة خلال بضع السنوات الأخيرة إلى درجة أصبح بالإمكان تطبيقها على تعرّف الصور حاسوبيًّا، وعلى إمكانات الذكاء الصنعي (AI) الأخرى. وقادنا استعمال وحدات المعالجة البيانية (GPU)، بدءًا من وحدة واحدة إلى مئات الوحدات، في شبكات تدريب عصبونية– تعمل بطريقة مشابهة للسيالات العصبية في الدماغ البشري– إلى تعرّف أنماطٍ في المعطيات لا يمكن تعرّفها في النظم الحاسوبية الأخرى. فالعقد الموجودة في طبقات الشبكات العصبونية يتعلم بعضها من بعض، وتتعلم من الشبكات العصبونية الأخرى، بطريقة تشبه كثيرًا تعلّم الأطفال. ومما يلفت الانتباه أن تعقيدها يجعل أيّ شخص عاجزًا عن معرفة كيفية تمكّن كل شبكة صنعية مدرَّبة من توليد نتائجها المفيدة.

إن التقدم السريع في الشبكات العصبونية والتعلم العميق هو نتيجة عوامل عدّة، منها ابتكار وحدات معالجة بيانية (GPUs) أفضل وأسرع من قبل، وتوفّر شبكات أوسع بطبقات أعمق، ووجود مجموعات بيانات واسعة وموسومة (عليها ملصقات) لتدريب الشبكات عليها، إضافة إلى أنواع جديدة ومتنوعة من الشبكات العصبونية والخوارزميات المحسَّنة. وبُغية تعرّف الصور حاسوبيا، وفّر الباحثون لنظام الشبكات العصبونية الكثيرَ من صور الأشياء –كالدراجات النارية والشمبانزي والأشجار والأشياء الفضائية، على سبيل المثال– لتتعلّم الشبكات العصبونية كيف تبدو الأشياء وكيف نميّزها من غيرها. فإذا درّب باحثٌ الشبكة لتتعرّف الحيوانات، فإن النظام سيتعلم بطريقة أسرع وأفضل إذا قدمنا له المعطيات القديمة لإنجاز المهمة الجديدة. وإذا كانت المهمة الأصلية، مثلًا، التمييز بين الأسود والحمير البرية فإن إضافة هذه المعطيات إلى مهمة التمييز بين الظباء والدببة يُعَدّ مساعدة لها.

ينجح النظام لأن لدينا الآن معرفة مشتركة بين مسارين، ويوضح ذلك الباحث كلون فيقول: "إذا كان نظام شبكة عصبونية بالأصل جيدًا في تعلّم مهمة فإنه سيكون أفضل وأسرع لدى تعلّم المهمة الثانية. يتفهّم النظام الأشياء المشتركة في المهمتين كالعيون والآذان والأرجل (القوائم) والفراء". وبتقدم التدريب تصبح الشبكة أحدّ ذكاء، وتغدو قادرة على تمييز الصور الفوتوغرافية وصور أخرى لم ترها من قبل. على سبيل المثال، أحرز كلون معدلات دقة تعرّف عالية باستعمال الشبكات

العصبونية، تصل إلى 96.6% مقارنة بما يزيد على 40,000 شخص تطوعوا لوضع لَصيقات (labels) على الصور نفسها. ويرى آخرون أن الشبكات العصبونية تتفوق على البشر بشكل ملحوظ. يتابع كلون فيقول: "في معظم الحالات يمكننا تدريب الشبكة العصبونية خلال يومين".

لا يعني هذا بالطبع أن كل النظم متساوية الفعالية– وأن النتائج متسقة الفائدة. إذ ينبغي ألاننسى أن الهدف توسعة حدود تعرف الصور حاسوبيا. يدرّب الباحثون اليوم النظم باستعمال اللصيقات. وهذا يعني تصميم صور لنوع واحد من الحيوانات "الأسد" ونوع آخر "حمار الوحش"، أو لمجرة "لولبية" وأخرى "إهليلجية". مُشْكلة هذا النهج أنه مستهلك للوقت وأحيانا مكلّف. أضف إلى ذلكأنك تفتقر أحيانا إلى لصيقات وقد تكون هذه مشوّشة، كما يقول سو زانغ (Ce Zhang) الأستاذ المساعد في مجموعة النظم لدى معهد ETH في زيوريخ. فاللصيقة cougar[2] قد تضلل النظام إذا عرضت مع أنواع الحيوانات والسيارات.

بالنتيجة، يهتم الباحثون بمجال بازغ للتعلم العميق، يعتمد على عدة طرائق تدريب، إضافة إلى التعلم بدون إشراف. بدأ الباحثون في الجامعات والشركات مثل Alphabet، التي تعمل على برمجيات Google Brain و DeepMind، بدراسة هذا المجال المعرفي. إنهم يعودون إلى النظم الترابطية (convolutional systems) التي تُنمذَج اعتمادًا على المعالجة البصرية عند البشر، وإلى النظم التوليدية (generative systems) التي تستند إلى نهج تقليدي، يعتمد على الإحصاء لتعلّم سمات مجموعات المعطيات.

فما هو الهدف النهائي؟ يقول شاونسكي: "نريد أن نسلِّم الحاسوب المعطياتِ والخوارزمية وعليه أن يسلمنا النتائج. سيُحدث هذا النوع من الإمكانات ثورةً في علم الفلك وكذلك في العلوم عامةً".

تركيز أشد

توسّع التطورات في الذكاء الصنعي حاليا حدود الشبكات العصبونية والتعلم العميق، ليشمل مجال الأمانة العلمية sci-fi[3]، إذ إن النتائج التي تنتجها هذه النظم حقيقية جدًّا. لنفترض: أن كلون يستعمل الآن نظمًا توليدية لإنتاج صور صنعية تبدو حقيقية للعين البشرية. تتناول هذه الصور الصنعية الحقيقية العصافير والحشرات وتتسع لتشمل الجبال والسيارات. إنه يصف هذه التقانة على أنها "مغيّر اللعبة". واللافت حقًّا فيها، هو أنه مع تقدم الزمن، تصبح بعض العصبونات في شبكات التعلم العميق أفضل من غيرها في تعرّف أشياء محددة وتوليدها، مثل العيون والأنوف أو البق والبراكين؟ ويتابع كلون قائلًا: "يكتشف النظام فعليا ما يُحتاج إليه للتعرف، ويعرّف العصبونات المتعلقة بهذه المفاهيم ويوضّعها آلياً".

من المؤكد أن للشبكات التوليدية قيمة تفوق إنتاج الصور الصنعية لأهداف الفن وألعاب الفيديو أو الحقيقة الموسعة/والحقيقة الافتراضية (AR/VR). لقد بدأ الباحثون يستعملون الشبكات التوليدية لتنافس شبكات تعرّف الصور، ولإعطاء نتائج أعلى دقة. ضمن هذا المشهد، تولد الشبكة المولدة صوراً مزيفة، على حين تقوم شبكة تعرّف الصور، المعروفة بالمميِّزة، بتحليل الصور ومحاولة فصل الصور الحقيقية عن الصور المزيفة. ثم تفحص الشبكة المميزة لاحقا صحة نتائجها،

[2] Cougar (الكوجر): حيوان نحيف الجسم كبير من صنف القطط. وهو حيوان ثديي آكل للحوم، لونه رمادي مائل إلى البني، يعيش في شمال أمريكا وجنوبها يسمى بالفرنسية Couguar وربما اشتق من اسمه Jaguar وهو اسم مشهور لنوع من السيارات. (المترجم)

[3] على غرار المصطلح "أمانة عالية (high fidelity) واختصاره hi-fi؛ نتكلم عن الأمانة العلمية (science fidelity) واختصاره sci-fi. (المترجم)

وتستعمل هذه النتائج لتجعل خوارزميتها أعلى دقة. وبمرور الوقت، تصبح الشبكة المميزة أشد ذكاء، وتخبر الشبكة المولّدة كيف تلائم خرجها لإنتاج صور أقرب إلى الحقيقية.

يقول زانغ: تكمن فائدة هذا النهج في أن الشبكة المميّزة التي يُشار إليها بالشبكة المولِّدة العدائية (Generative adversarial network (GAN))، تتعلم بمرور الوقت، الأمور الأكثر أهمية في الصورة. ويتابع زانغ، عند نقطة معينة، يظهر النظام حدساً قريباً من الحدس البشري: "إذ تتحسن النتائج تحسناً لافتًا". والأمر المهم، هو أن هذا النهج لا يحسّن كشف الصور فحسب، وإنما يمكنّه أن يختصر أيضا الوقت اللازم لتدريب الشبكة بخفض عدد الصور –حجم المعطيات الأساسي– الذي نتطلبه للحصول على نتائج مفيدة. يتابع زانغ: "ثمة مسألة مهمة وهي: كيف يمكننا خفض متطلبات الشبكة العصبونية أي حجم المعطيات التي تحتاج إليها لإنجاز مستوى النوعية الحالي؟".

وهناك خطوة أخرى تُعنى في جعل الشبكات العصبونية الحالية أسهل استعمالا. مازالت التقانة في بداياتها ويكافح الباحثون غالبا لاستعمال الأدوات والتقانات على نحو فعال. إذ عليهم في بعض الحالات العمل على عدة شبكات بطريقة تفاعلية لإيجاد الشبكة التي تعمل على نحوٍ أفضل. وبالنتيجة، طوّر زانغ برمجيات أطلق عليها اسم: easy.ml، تجعل شبكات التعلم العميق العصبونية أكثر آلية وفعالية. وهي تتضمن مكوّنات مستمثَلَة مثل وحدات المعالجة المركزية (CPUs)، ووحدات المعالجة البيانية (GPUs) وصفيفات البوابات القابلة للبرمجة حقلياً (FPGAs)، إضافة إلى لغة تصريحية (declarative language) لإدارة الخوارزميات على نحو أفضل.

ويتابع زانغ فيقول: "يحتاج المستثمر حالياً أن يتعامل مع تنوّع أكثر من القرارات ومنها نوع الشبكة العصبونية التي عليه استعمالها. قد يوجد 20 نوعًا مختلفًا من الشبكات العصبونية المتاحة للقيام بالمهمة ذاتها. ومن المهم اختيار النموذج الصحيح وخفض التعقيد".

تنضم البرمجيات حاليا إلى تقانات أخرى للتعلم العميق– ومنها خوارزمية تسمى ZipML تخفض تمثيل المعطيات دون أن تخفض دقتها– وقد حذفت الضجيج وجعلت الصور أكثر وضوحا، على نحو ملموس، لدى مجموعة الفضاء في معهد ETH في زيوريخ. وهكذا، يستطيع شاونسكي وآخرون استكشاف الكون بعمق أكثر.

ويشرح باحثون من معهد ETH في زيوريخ: "لا تشبه حالتنا مجالات العلم الأخرى، إذ لا نستطيع إجراء التجارب في المختبرات ومن ثمّ تحليل النتائج ببساطة، نحن نعتمد على المقاريب وعلى الصور للنظر إلى الوراء في الزمن. علينا أن نجمّع معًا كل هذه اللقطات الثابتة –والتي تشكل أساساً مجموعات معطيات ضخمة– للحصول على المعرفة والنظرة الأعمق".

ويضيف الباحث لانوس: "ستحل قريبًا علوم المعطيات والتعلم الآلي محل الطرائق التقليدية في علم الفلك والفضاء. وهي لاتقوم بالعمل بطريقة أفضل فحسب، وإنما تقدم أيضا طرائق جديدة للنظر إلى المعطيات".

إن النظر في المستقبل مهمّ أيضاً. ويقول لانوس، ستقود الشبكات العصبونية، في السنوات القادمة إلى تطورات هائلة في مجالات أبعد من الفيزياء الفلكية. فلن تكشف هذه النظم الأشياء وتتعرّفها وتصنفها فحسب، وإنما ستدرك ما يحدث في الصور والمشاهد بالزمن الحقيقي. وهذا بالطبع سيؤثر عميقًا في كل شيء، بدءاً من طريقة قيادة العربات الآلية إلى مهامّ التشخيص الطبي. وفي النهاية، ستساعدنا هذه النظم على فك أسرار كوكبنا، بل والكون. وستقدم لنا مستوى فهمٍأعلى، لم يكن متصوراً قبل بضع سنين.

يقول لانوس: "يتطور تعرّف الصور حاسوبيًا تطورًا سريعًا. لقد وجدنا طرقاً أسرع وأفضل لتدريب الشبكات. وإن كل كسب في السرعة أو في الدقة ولو كان بنسبة قليلة هو فرقٌ عميق الأثر في عالمنا الحقيقي".

مراجع للاستزادة

- Nguyen, A., Yosinski, J., Bengio, Y., Dosovitskiy, A., and Clune, J. "Plug & Play Generative Networks: Conditional Iterative Generation of Images in Latent Space". Computer Vision and Pattern Recognition (CVPR '17), 2017. http://www.evolvingai.org/ppgn
- Lanusse, F., Quanbin, M, Li, N., Collett, T.E., Li, C., Ravanbakhsh, S., Mandelbaun, R., and Poczos, B. "CMU DeepLens: Deep Learning for Automatic Image-based Galaxy-Galaxy Strong Lens Finding". March 2017. arXiv:1703.02642. https://arxiv.org/abs/1703.02642.
- Wang, K., Guo, P., Luo, A., Xin, X., and Duan, F. "Deep neural networks with local connectivity and its application to astronomical spectral data". 2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC), Budapest, 2016, pp. 002687-002692. doi: 10.1109/SMC.2016.7844646. http://ieeexplore.ieee.org/ document/7844646/
- Goodfellow, I.J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. "Generative Adversarial Networks". June 2014. eprint arXiv:1406.2661. http://adsabs.harvard.edu/cgi-bin/bib_query?arXiv:1406.2661.

المؤلف

صموئيل غرينغارد (Samuel Greengard) هو كاتب وصحفي في West Linn بولاية أوريغون Oregon الأمريكية.

التفكير العميق لتوليدِ كلامٍ أجود

THINKING DEEPLY TO MAKE BETTER SPEECH*

Neil Savage
ترجمة: م. سماح راغب
مراجعة: د. أميمة الدكاك

ثمّة حاجة إلى مزيد من العمل لجعل الكلام المركّب طبيعيًّا أكثر، وأسهل فهماً، وأكثر إمتاعاً عند سماعه.

ليست الآلات الناطقة بالشيء الجديد، فمنذ عام 2011 تجيب Siri عن أسئلة يطرحها مستعملو iPhone، وبرمجيات تحويل النص المكتوب إلى الكلام المنطوق منتشرة منذ مدة أبعد.

وقد استعمل الأشخاص ذوو الإعاقات النُطقية – وأشهرهم ستيفن هوكينغ (Stephen Hawking)– الحواسيب لتوليد الكلام المنطوق عقوداً من الزمن. ومع ذلك، فإن تركيب الكلام بحيث يبدو طبيعيًّا كما لو أن إنساناً ينطق به ما يزال هدفاً بعيد المنال، مع أنه يبدو كأنه شارَفَ على أن يصبح واقعيًّا.

يقول Simon King وهو أستاذ معالجة الكلام ومدير مركز بحوث تقانات الكلام في جامعة أدنبرة: "إذا استمعت إلى الإصدار الأخير لبرنامج Siri الخاص بـ Apple فسيبدو الأمر مدهشاً". لدى كلٍّ من Apple و Google و Microsoft تطبيقات تجارية لقراءة النصوص المكتوبة بنبرة محايدة ولكنها مقبولة. تنطق، هذه البرامج، معظم الكلمات نُطقاً صحيحاً، وتنساب عموماً من كلمة إلى الكلمة التالية في جُمل مقبولة تماماً. يقول King: "نحن بالنهاية جيدون في هذا إلى حدٍّ بعيد، والكلام مفهوم جدًّا".

ومع ذلك، يرغب الباحثون في تركيب الكلام، في الذهاب إلى ما هو أبعد من مجرد الكلام "المفهوم"، إلى كلام طبيعيٍّ أكثر. يمكن لعملهم أن يجعل الكلام المركّب أسهل فهماً وأكثر إمتاعاً عند سماعه، كما يمكن أن يسمح لهم بتركيب أصوات أفضل للأشخاص غير القادرين على التحدث بأنفسهم، وإنشاء نظم لتحويل النص المكتوب إلى الكلام المنطوق تخصّ لغات أقل شيوعاً.

يقول Alex Acero، كبير مديري Siri في شركة Apple: "تعمل جميع النظم بصورة جيدة عمليًّا على مستوى الجُمل". ولكن، اطلب من الآلة أن تقرأ لك مقالاً صحفيًّا أو رسالة بريد إلكتروني من والدتك وستكون النتيجة مسطحة (بدون تنغيم[1]). ويتابع: "نعم، يمكنك فهمها إذا أصغيت إليها، لكنها ما تزال مختلفة عمّا إذا قرأها لك شخص ما". لا يستطيع الكلام الحاسوبي

**نُشِر هذا البحث في مجلة Communications of the ACM، المجلد 60، العدد 3، آذار (مارس) 2017، الصفحات 15 – 17.*

[1] التنغيم (prosody) هو تغيرات التردد الأساسي (تواتر اهتزاز الأوتار الصوتية أثناء الكلام) وحدّة الصوت (ارتفاعه) ومدد مختلف الصوتيمات (phonemes) مع الزمن. هذه التغيرات هي التي تميز الكلام البشري، وهي التي تسعى مركّبات الكلام إلى إعادة إنتاجها. (المترجم)

ربوط شبيه بالإنسان أطلق عليه مبتكروه من Toshiba وجامعة Osaka اسم Aiko Chihira أثناء تجربةٍ جرت في طوكيو عام 2015 في متجر Mitsukoshi. تقول Toshiba إنها ستدمج تركيب الكلام وتعرُّفَه ضمن الربوط بحلول عام 2020.

التعامل مع التنغيم (الإيقاع ونبرة الكلام[2] التي توصِل المعنى وتضيف السياق العاطفي). يقول Acero: "هذا أمر مهم جدًّا بالنسبة للبشر، ولهذا السبب، فإنك تضيف رموزاً تعبيرية عندما ترسل رسائل نصية".

ثمة نهجان أساسيان لإنشاء الكلام. أقدمهما هو تركيب الكلام باستعمال الموسطات، حيث يولّد فيه الحاسوبُ الأصوات انطلاقاً من عناصر النص. تطور ذلك على مر السنين ليصبح تركيب الكلام معتمداً على موسطات إحصائيّة، وهو يستعمل نموذجاً إحصائيًّا لإنشاء شكل الموجة المناسب لكل صوت. لقد كان النموذج الإحصائي المستعمل مدة طويلة هو نموذج ماركوف المخفي الذي يحسب الحالة المستقبلية للنظام اعتماداً على حالته الراهنة. غير أنه في العامين الماضيين، استُعيض عن نماذج ماركوف المخفية بالشبكات العصبونية العميقة التي تحسب التفاعل بين عوامل مختلفة في الطبقات المتتالية. يقول King إن هذا التحول قد أدى إلى تحسن في دقة النهج المعتمد على الموسطات.

إنّ التقنية التي جرى استعمالها في المقام الأول على مدار العقدين الأخيرين في تركيب الكلام هي السَّلسَلة (الضم) (concatenative speech synthesis)، وفيها يسجل شخص ما، عدة ساعات من الكلام، يجري بعد ذلك تقطيعه

[2] نبرة الكلام (intonation) هي تغيرات التردد الأساسي مع الزمن. (المترجم)

إلى وحدات صوتية مفردة تسمى صوتيمات[3] (phonemes)، ومن ثم يجري ربطها معاً لإنشاء عبارات جديدة لم يلفظها المتحدث الأصلي البتّة. على سبيل المثال، يقسم مركّب الكلام من Apple الصوتيمات، الممَثلة على شكل موجة إلى نصفين. ويوضح Acero أن ذلك يتيح المزيد من الخيارات لإيجاد الصوتيمات المختلفة التي تتلاءم معاً بسلاسة.

يضم الإصدار الأخير من Siri تركيب الكلام المعتمد على الموسطات إلى تركيب الكلام بالسَلسلة. إنه يعتمد على نموذج إحصائي يسمى شبكة الكثافة المختلطة (mixed density network) –وهو نوع من الشبكات العصبونية – التي تتعلم موسطات الصوتيمات التي تبحث عنها، متفحّصة المئات من السمات من مثل: هل كان الصوت منبوراً أم لا، أيُّ الصوتيمات تسبق عادةً الصوتيمات الأخرى أو أيّها تتبعها. ما إن يُعرف الشكل الموجي الذي يُفترض أن يبدو عليه الكلام حتى يبحث عن الأشكال المناسبة في الكلام المسجل ويلائمها معاً. لا يُنشئ النظام بالضرورة كل عبارة من الصفر؛ إذ يمكن أن تؤخذ مجموعات من الكلمات وأحياناً جُمل بكمالها من التسجيل مباشرة. يقول Acero: "هذا النهج مؤتمت بطريقة أفضل، وهو أكثر دقة لأنه مَقود بالبيانات بدرجة أكبر".

ومع أن النتائج جيدة جداً فإن الكلام ما يزال يفتقر إلى التنغيم لأن الآلة لا تفهم حقًّا ما تقوله. يعتقد King أن هذا القصور قد يفسر إحدى مشكلات الكلام المركّب، ففي الوقت الذي قد يكون فيه الكلام مفهوماً تماماً لشخص ما يصغي إليه في غرفة هادئة، تنخفض قدرة الفهم بسرعة أكبر بكثير مما يحدث مع الكلام الطبيعي حين يكون المستمع في بيئة مفعمة بالضجيج، أو يحاول القيام بمهام متعددة، أو كان يعاني من ضعف السمع أو صعوبة في الفهم.

يفترض King أن هذا التدهور يحدث لأن الكلام الطبيعي يحتوي على الكثير من الحشو والتلاميح التي تساعد على فهم ما يقال. ففي الكلام الطبيعي، قد تكون هناك، على سبيل المثال، تغيرات في طبقة الصوت أو نبرته أو نغمته، عندما تُفضي كلمة إلى كلمة أخرى. هذه التلميحات الصوتية ليست موجودة في الكلام المركّب، وفي الكلام المسلسل قد تحتوي الكلمات المأخوذة من جمل مختلفة على تلميحات خاطئة أيضاً.

وقد يحدث أيضاً أن يؤدي الاضطرار إلى معالجة حالات عدم الاتساق هذه، إلى دفع دماغ المستمع للعمل بجد أكثر، مما قد يؤدي إلى فقدان شيء ما. يقول King: "لا يمكنك أن تقول إن كلامك المركّب طبيعي حقًّا ما لم يصبح بجودة الكلام الطبيعي بالنسبة للجميع، وفي كل الأوساط".

يتابع King: "لكي تقول شيئاً بطريقة أقرب ما تكون إلى الكلام الطبيعي، فأنت بحاجة إلى فهم معناه". ويضيف: "مع أن تعرّف الكلام في Siri والنظم المماثلة، جيدٌ بقدرٍ كافٍ للرد على الأسئلة والاستجابة للأوامر، فإن مستوى فهمها ما زال ضحلاً إلى حدٍّ ما". يمكنها تعرّف الكلمات المعزولة، وتحديد الأسماء والأفعال، وملاحظة بنية الجملة المحلية، وحتى تمييز الأسئلة من العِبارات. يستعمل الباحثون الذين يعملون في فهم اللغات الطبيعية نُهُجاً (طرائقَ) مثل الفضاءات الشعاعية التي تركز على إحصائيات مثل تواتر ظهور الكلمات، ولكن إلى الآن لا تستطيع الآلات فهم الكلام – خاصة في الأجزاء الكبيرة مثل الفقرات أو المقاطع الكاملة– بمستوى من العمق كافٍ لجعلها قادرة على قراءته كما يقرؤه الإنسان.

[3] الصوتيم هو الوحدة الصوتية الأساسية في إنتاج الكلام. منها الصوائت (vowels) مثل أصوات حروف المد والفتحة والضمة والكسرة، ومنها الصوامت أو السواكن وهي أصوات حروف الهجاء بوضع صوت الهمزة مكان الألف وأصوات الواو والياء غير المدِّية مكان هذين الحرفين.

موجة جديدة

أعلنت شركة Google في أيلول من عام 2016 أنها قطعت أشواطاً كبيرة في تقنية تدعى WaveNet طوّرتها شركة DeepMind وهي شركة مقرها لندن اشترتها Google في عام 2014. تستعمل WaveNet تركيباً إحصائيًّا للموسطات يعتمد على الشبكات العصبونية العميقة لإنتاج كلام باللغتين الإنكليزية والماندرين صنّفه المستمعون على أنه أحسن من أفضل نُظم التركيب الموجودة (لا يوجد قياس موضوعي لجودة الكلام، لذا يجري تقويمها دائماً من قِبل المستمعين). كما ولّد النظام آليًّا موسيقا لآلة البيانو. لقد نشرت Google نتائجها ضمن مدونة وفي ورقة بحثية على موقع ArXiv، لكنها رفضت السماح للباحثين بإجراء مقابلات صحفية.

استوحت Google نهجها من نموذج كانت قد نشرته في وقت سابق من نفس العام الذي استعملت فيه شبكة عصبونية لتوليد صور ذات مظهر طبيعي، بكسل واحد في كل مرّة. درّب الباحثون النظام عن طريق تغذيته بتسجيلات لمتحدثين بشر. يمكن أن يحتوي مثل هذا التسجيل الصوتي الخام على 16,000 عينة في الثانية، لذا فهو مُكلِّف حسابيًّا. بعد أن انتهى تدريب النظام، غَذَّوه بِنَص كانوا قد قسموه إلى سلسلة من السمات اللغوية والصوتية، مما أعطى الحاسوب معلومات من قبيل ما هي الكلمة والمقطع[4] (syllable) والصوتيم التي كان يراها. لقد تمكنوا من تدريبه على أصوات متحدثين مختلفين كي يستطيع التحدث بأصوات مختلفة، وزوّدوه بلهجات وعواطف مختلفة.

يصف Acero تقنية WaveNet بأنها نهج هام جدًّا قد يحل مستقبلاً محل التركيب المسلسل. يستغرق إنتاج ثانية واحدة من الكلام في الوقت الراهن عدة ساعات من الحوسبة، لذا فهو ليس عمليًّا فوراً.

نموذج فيزيائي

يتّبع Oriol Guasch الفيزيائي والرياضي في جامعة Ramon Llull في برشلونة – إسبانيا، أيضاً نهجاً مكثفاً حسابياً لتركيب الكلام، وهو يعمل على نمذجةٍ رياضيةٍ لكامل الجهاز الصوتي البشري. ويقول: "نَودُّ محاكاة العملية الطبيعية برُمّتها، وهذا الأمر سيؤدي في آخر المطاف إلى توليد الصوت النهائي".

للقيام بذلك، يأخذ صورة الرنين المغناطيسي للجهاز الصوتي لشخص ما، أثناء نطقه الصائت[5] "E" على سبيل المثال، ثم يمثل الأشكال الهندسية للحبال الصوتية، والحنك اللحمي، والشفتين، والأنف، والأجزاء الأخرى بمعادلات تفاضلية. واعتماداً على تلك المعادلات يولد شبكة حاسوبية – شبكة متعددة الجوانب تُقارب الشكل الهندسي. يقول Guasch: "العملية ليست بالسهلة إذ يمكن للحاسوب المكتبي أن يولد شبكة تتكون من ثلاثة إلى أربعة ملايين عنصر في غضون ثلاث أو أربع ساعات لتمثيل صوت الفتحة". ومع ذلك، يتطلب لفظ حرف الصفير "S" عمل جهاز حاسوب يضم 1000 معالج مدة أسبوع لتوليد 45 مليون عنصر. ينشأ التعقيد الإضافي لذلك الصوت من تدفق الهواء بين الأسنان وتوليد دوّامات عنيفة تحوم في أنماط معقدة. تخيل إذن الوقت اللازم لإنتاج كلمة كاملة، قضلاً على جملة.

[4] المقطع الصوتي جزء من الكلمة يتضمن صائتاً (vowel) واحداً. (المترجم)

[5] الصائت (vowel) في لغتنا العربية هو صوت أحد حروف المد أو الحركات. (المترجم)

يعتبر Guasch نهجه تحديًّا حسابيًّا مثيراً للاهتمام أكثر منه محاولة عملية لإنشاء الكلام. ويقول: "لا يقتصر الهدف النهائي على تركيب الكلام وحسب، وإنما يتمثل في إعادة إنتاج الطريقة التي يتصرف بها جسم الإنسان"، ويضيف: "أعتقد أنه عندما تعترضك مشكلة حسابية، فمن الجيد التصدي لها من زوايا مختلفة".

من ناحية أخرى، يعمل King من جامعة إدنبرة في مجال التطبيقات العملية. وقد حصل أخيراً على تمويل لمشروع يمتد على ثلاث سنوات بالاشتراك مع BBC World Service لإنشاء نُظم لتحويل النص المكتوب إلى كلام منطوق تخص اللغات التي لا يتكلمها عددٌ كافٍ من الناطقين، ليكون تطوير النظام عامل جذب مالي للشركات. ويقول King: "يجب أن يكون بالإمكان استعمال التعلم الآلي على بيانات مثل البث الإذاعي والصحف لبناء نظام موثوق به دون الإنفاق على توظيف خبراء لغويين وفناني صوت محترفين". لقد بنى بالفعل نموذجاً للغة السواحلية يقول إنه يعمل بصورة جيدة.

كما طوّر King نظاماً يمكنه أخذ عينة صغيرة من تسجيلات كلام فرد معين وتطبيقها على نموذج دُرِّب بالفعل على مجموعة بيانات أكبر، واستعمالها لتوليد كلام جديد يبدو كأنه لهذا الشخص. يخضع النظام للتجارب السريرية في مستشفًى في المملكة المتحدة لِيُعرَف: أيمثّل طريقة عملية لمساعدة الأشخاص المصابين بالتصلب الجانبي الضموري، الذين من المتوقع أن يفقدوا قدرتهم على الكلام مع تقدم مرضهم. ويقول: "لن يساعدهم ذلك على العيش مدة أطول، ولكن يمكن أن يساعد على تحسين طبيعة حياتهم في الوقت الذي يعيشون فيه".

لقراءة المزيد

- *Van den Oord, A., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., and Kalchbrenner, N.* WaveNet: A Generative Model for Raw Audio, ArXiv, Cornell University Library, 2016 http://arxiv.org/pdf/1609.03499
- *King, S., and Karaiskos, V.* The Blizzard Challenge 2016, Blizzard Challenge Workshop, Sept. 2016, Cupertino, CA, http://www.festvox.org/blizzard/bc2016/blizzard2016_overview_paper.pdf
- *Arnela, M., Dabbaghchian, S., Blandin, R., Guasch, O., Engwall, O., Van Hirtum, A., and Pelorson, X.* Influence of vocal tract geometry simplifications on the numerical simulation of vowel sounds, *Journal of the Acoustical Society of America*, 140, 2016, http://dx.doi.org/10.1121/1.4962488
- *Deng, L., Li, J., Huang, J-T., Yao, K., Yu, D., Seide, F., Seltzer, M., Zweig, G., He, X., Williams, J., Gong, Y, and Acero, A.* Recent advances in deep learning for speech research at Microsoft, IEEE International Conference on Acoustics, Speech and Signal Processing, 2013, http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6639345
- Simon King – Using Speech Synthesis to Give Everyone Their Own Voice, https://www.youtube.com/watch?v=xzLpxcpo-E

مستقبل أنصاف النواقل

THE FUTURE OF SEMICONDUCTORS*

Samuel Greengard
ترجمة: د. أحمد حصري
مراجعة: د. خالد مصري

يجتهد الباحثون لإيجاد طرقٍ جديدة لتقدُّم أنصاف النواقل مع اقتراب قانون مور من حدوده النهائية.

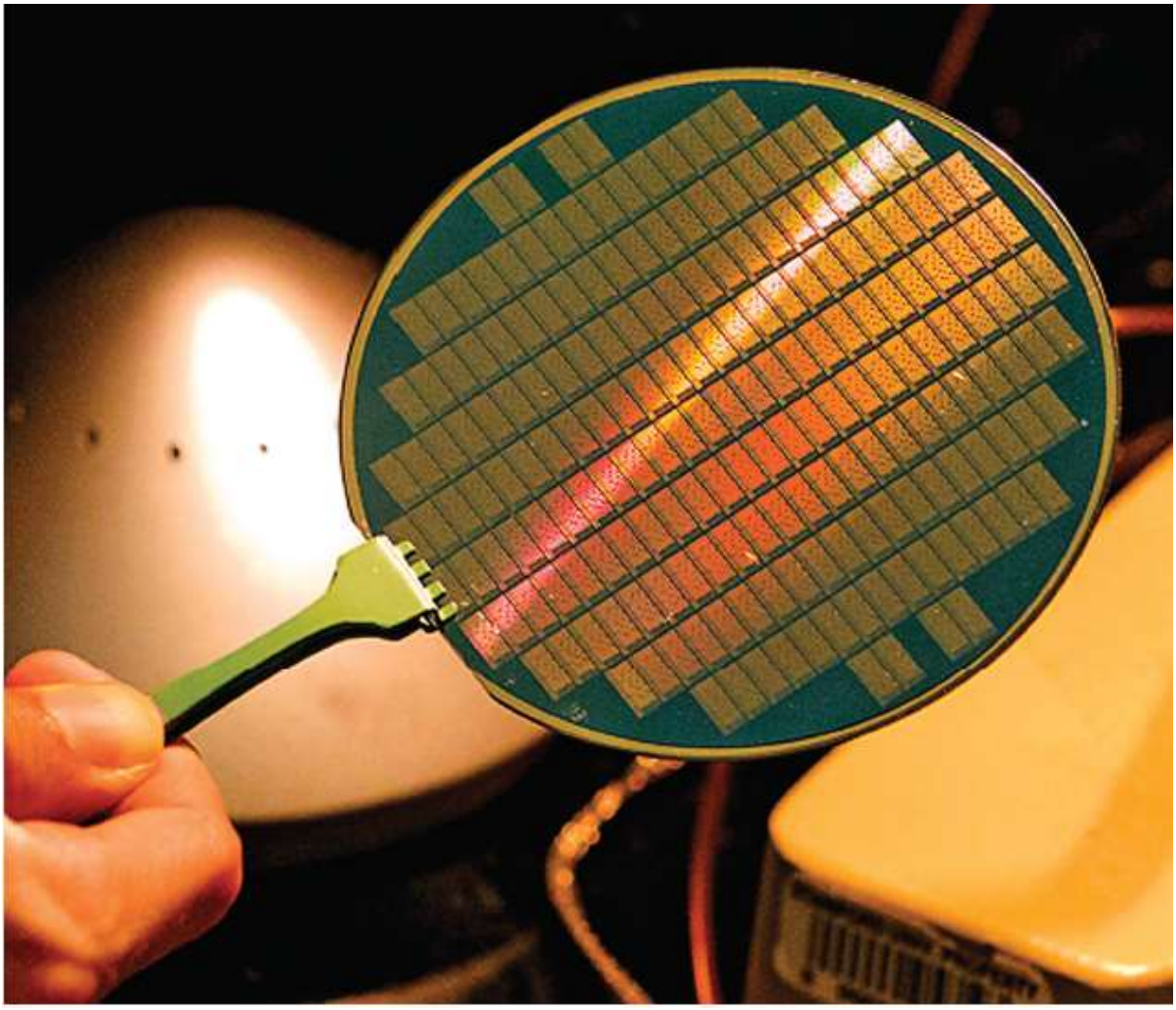
لقد بقي قانون مور (Moore's law) على حاله إلى حدٍّ ما، أثناء نصف القرن المنصرم، مع أن الحوسبة تقدمت تقدماً كبيراً. يعتمد مفهوم مور المنسوب إلى غوردون مور (Gordon Moore) الرائد في مجال أنصاف النواقل، على ملاحظةٍ مفادها أن عدد الترانزستورات التي يمكن حشدها في دارة متكاملة (Integrated Circuit, IC) يتضاعف مرتين تقريباً كل عامين. وقد بقي هذا المفهوم أكثر من خمسين عاماً يقدم إطارَ عملٍ للتنبؤات المُتعلقة بتطور أنصاف النواقل. وكان أن ساعد مفهوم مور مصنِّعي الحواسيب وشركاتٍ أخرى عديدة على تركيز بحوثهم وخططهم المستقبلية.

مع هذا، توجد مؤشرات إلى أن قانون مور بلغ نهاية مساره العملي. ومع أن صناعة الدارات المتكاملة ستستمر بإنتاج ترانزستوراتٍ أسرع وأصغر حجماً أثناء السنوات القليلة القادمة، فإن هذه النُظُم لن تستطيع العمل عند الترددات المُثلى، بسبب قضايا التبديد الحراري. هذا ما "جعل وتيرةَ سرعة تقدّم أداء الحوسبة صغيرةً"، كما أشار إلى ذلك زميلا IEEE توماس كونت (Thomas Conte) و باولو غارجيني (Paolo Gargini) في تقريرهما IEEE-RC-ITRS في العام 2015، "*أساس صناعة الحوسبة الجديدة بعد العام 2020*" (*On the Foundation of the New Computing Industry Beyond 2020*).

إلَّا أن التحدِّيات لا تقف عند هذا الحدّ. فهناك أيضاً حقيقة وهي أن الباحثين لا يستطيعون الاستمرار بتصغير تصاميم الجذاذات (chips)؛ ففي مرحلةٍ ما أثناء السنوات القليلة القادمة، ستبلغ الدارات المتكاملة الثنائية البعد الحالية الأبعادَ الحدِّية العملية. ومع أن الباحثين يجرون تجاربَ بموادٍ وتصاميم جديدةٍ– بعضها مختلف جذرياً– فلا يوجد مسارٌ واضحٌ للتقدّم حالياً. تنبأ مور عام 2015: أن القانون الذي يحمل اسمه سيخبو خلال عقدٍ من الزمن. أشار التقرير IEEE-RC-ITRS إلى "أن هناك حاجةً مُلحّة لطريقة حوسبةٍ جديدة."

نتيجةً لما سبق، فإن صناعة أنصاف النواقل في حالةٍ من التغير المستمر. هناك اعترافٌ متنامٍ بأن على البحث والتطوير أن يدمج تصاميمَ جديدة للدارات، وأن يعتمد على طرائق مختلفة تماماً لرفع قدرة الحوسبة بمقدارٍ أكبر. "لقد مرّ على المهندسين عدة سنوات لم يحتاجوا خلالها لبذل جهودٍ كبيرةٍ لرفع الأداء والوظيفية،" هذا ما لاحظه جان راباي (Jan Rabaey) الأستاذ ورئيس شعبة الهندسة الكهربائية (EE) في قسم الهندسة الكهربائية وعلوم الحاسوب في جامعة كاليفورنيا بولاية بيركلي. وأضاف "عندما نصل إلى الحدود الفيزيائية للتقانات الحالية، فإن الأمور تغدو أشدّ صعوبةً بكثير."

**نُشِر هذا البحث في مجلة Communications of the ACM، المجلد 60، العدد 3، آذار (مارس) 2017، الصفحات 18 – 20.*



ابتكر المهندسون رقائق كهذه من الأنابيب الكربونية النانوية، وهي مرشحةٌ لتحل محل رقائق السليكون ولتوفر معالجات صغرية أصغر حجماً وذات كفاءة طاقية أفضل.

الترانزستور المتقلِّص إلى حدٍ لا يصدق

لقد اتَّبع تاريخ أنصاف النواقل وقانون مور مساراً طويلاً ومتعرجاً. أشار كونت (Conte)، وهو أستاذٌ في مدارس علم الحاسوب وهندسة الحاسوب في معهد جورجيا للتقانة، إلى أنه لم تُربَط الحوسبة دائماً بتقليص الترانزستورات. وأشار إلى أن " عمر الظاهرة لا يزيد عن ثلاثة عقود." قبل سبعينيات القرن المنصرم، جرى بناء الحواسيب الفائقة الأداء من أمثال Cary-1،

باستعمال عناصرَ منفصلة معتمدة على منطق الباعث المرتبط[1]. وأشار إلى أن " في الواقع، لم يبدأ أداء المعالجات الصغرية وكلفتها في بَزِّ هذه التقانات (أيْ التفوق عليها)، إلّا في منتصف الثمانينيات من القرن العشرين."

عندها بدأ المهندسون الذين يطورون النُظُم العالية الأداء، بالانجذاب نحو قانون مور، ويعتمدون التركيز على المعالجات الصغرية. إلّا أن العائدات الكبيرة لم تدم طويلاً. ففي منتصف التسعينيات من القرن الماضي، أشار كونت إلى أن "التأخيرات في الأسلاك على الجذاذة فاقت التأخيرات الناجمة عن سرعات الترانزستورات". قاد ذلك إلى ما يُسمى "جدار التأخير السلكي"، وقد تجاوز المهندسون الأمر باستعمال التوازي بشكلٍ غير مرئي (وراء مسرح الأحداث). ببساطة: تستخلص التقانة التعليمات وتنفذها على التوازي، ولكن على شكل مجموعاتٍ مستقلة. وقد أُطلق على ذلك اسم " العهد السُلَّمي الفائق " (superscalar era). ومع أن المعالج الصغري بنتيوم برو (Pentium Pro) ليس أول نظام استعمل هذه الطريقة، فإنه برهن على نجاح هذه المنهجية.

واجه المهندسون في منتصف العقد الأول من القرن الواحد والعشرين جدارَ القدرة. فلأن قدرة الترانزستورات من النوع CMOS (معدن–أكسيد–نصف ناقل متتامين Complementary metal–oxide–semiconductor) تتناسب مع تردد تشغيلها، فإن تبريدها يصبح إلزامياً، عندما تصل كثافة القدرة إلى $200\ W/cm^2$. يُفسِّر كونت ذلك بالقول "يمكنك تبريد المنظومة، غير أن تبريد شيء لتبديد استطاعة تقارب 150 W يشبه تابعاً خطوياً، لأن الاستطاعة 150 W قريبةٌ من حدّ تقانة التبريد بالهواء القسري الرخيصة نسبياً". لا يتطور استهلاك الطاقة والأداء بنفس الطريقة. ويضيف كونت " لقد كنا نُخفي هذه المسألة عن المبرمجين. ولكننا لا نستطيع الآن القيام بذلك مع CMOS."

لم يعد بإمكان المهندسين حشد عددٍ أكبر من الترانزستورات على رقاقة (wafer) مع الاحتفاظ بالكسب ذاته. وهذا ما دفع إلى تخفيض تردد نواة المعالج وإدخال معالجات متعددة النوى. ومع ذلك، فقد بقيت المشكلة قائمةً. فمع تصغير أبعاد الترانزستورات–وصلت إلى نحو 65 nm في عام 2006– استمرت المكاسب الاقتصادية والأداء بالهبوط تبعاً لذلك، ومع هبوط العُقَد إلى 22 nm و 14 nm ازدادت المشكلة سوءًا.

أكثر من ذلك، كل ما سبق ساهم في زيادة تكلفة بناء مرافق التصنيع إلى حدٍّ غير معقول، وأصبح تصنيع أنصاف النواقل مفرطَ التكلفة. يوجد حالياً أربعة مصانع عالمية فقط لأنصاف النواقل هي: Intel و TSMC و GlobalFoundries و Samsung. وهذا أقل من قرابة دَسْتَتَيْن (dozen) كانت موجودة قبل عقدين من الزمن.

من المؤكد أن صناعة أنصاف النواقل اقتربت من الحدود الفيزيائية للترانزستورات CMOS. ومع أن هناك تقانات بديلة في مرحلة البحث والتطوير حالياً، تشمل الأنابيب الكربونية النانوية وترانزستورات مفعول الحقل النفقي (tunneling field effect transistors, TFET)، فلا يوجد دليلٌ على أن هذه الأجيال الجديدة من التقانات ستدخل حيز التنفيذ بطريقة رئيسية. ومع أنها تبشر بتحقيق المزيد من الكسب في الأداء، فإنها تستطيع في أحسن الأحوال إطالة أمد قانون مور بجيلٍ أو جيلين.

في الحقيقة، أعلنت مبادرة مجموعات صناعية مثل IEEE International Roadmap of devices (IRDS) أنه سيكون من المستحيل تقريباً تقليص حجم الترانزستورات أكثر بحلول العام 2023.

[1] الدارات المنطقية ذات الباعث المرتبط (emitter-coupled logic, ECL) هي دارات متكاملة عالية السرعة من الطائفة المنطقية للترانزستورات الثنائية القطبية. تستعمل مضخماً تفاضلياً من الترانزستورات الثنائية القطبية (overdriven). بمدخلٍ أحادي (single-ended) وتيار باعث محدود لتلافي العمل في منطقة الإشباع وسلوكها البطيء عند الإطفاء (turn-off). (المراجع)

أعلن ميشال شودزيك (Michael Chudzik) وهو مدير كبير في مؤسسة المواد التطبيقية (Applied Materials) "أن تقانة أنصاف النواقل تواجه تحدياتٍ على جبهاتٍ عديدة. منها تحدياتٌ هندسية وأخرى اقتصادية بسبب انخفاض عدد الصناعيين، إضافةً إلى التغيرات الأساسية في طريقة استعمال الناس لتجهيزات الحوسبة"، مثل الهواتف الذكية، إضافةً إلى الحوسبة السحابية (cloud computing) وإنترنت الأشياء (IoT)، التي تختلف متطلباتها من الدارات المتكاملة اختلافاً كلياً. وهذا ما يجعل طرائق الماضي غير مرغوبٍ فيها مستقبلاً. يقول رباي (Rabaey) " إننا ندخل عصراً مختلفاً".

تصاميم المستقبل

إن التخطيطَ لمستقبل الدارات المتكاملة والحوسبة أمرٌ بالغ الأهمية. يقول شودزيك، يكمن أحد خيارات تطوير أداء الجذاذة في استعمال مواد مختلفة. على سبيل المثال، يُجرِّب الباحثون الكوبالت ليحلّ محلّ التنغستين والنحاس لزيادة حجم الأسلاك، ويدرسون مواد بديلة للسليكون Si. يشمل ذلك الجرمانيوم (Ge) والخليط SiGe و المواد III-V مثل زرنيخ الغاليوم وزرنيخ الإنديوم غاليوم. إلّا أن هذه المواد تعاني من تحديات تتعلق بأدائها واستعمالها على نطاقٍ واسع. وحتى لو أمكن معالجة هذين الأمرين فإن الكسب الناجم سيكون ضئيلاً، ولا يلبث أن يتلاشى خلال زمنٍ قصير.

بمواجهة نهاية قانون مور، ركّز الباحثون اهتمامهم على منهجياتٍ جديدة ومختلفة تماماً. يكمن أحد الخيارات الواعدة بتكديس المكوّنات والتوسع من التصميم الثنائي البعد الحالي للدارات المتكاملة إلى ثلاثي الأبعاد، ربما باستعمال أسلاك نانوية. يقول رباي "إن الانتقال إلى البعد الثالث إضافةً إلى تكديس الذاكرة والبوابات المنطقية، يُمكِّن من تحقيق وظائف أكثر بكثير بواحدة الحجم." ومع ذلك، فإن تصميم الجذاذات الثلاثية الأبعاد تعاني حالياً من تحدياتٍ، خاصةً فيما يتعلق بتبريدها. فللتجهيزات الناتجة سطح حجمٍ أصغر بفعل تكديس المهندسين للمكوّنات. نتيجةً لذلك، " يجب اللجوء فجأةً إلى المعالجة عند درجات حرارة أخفض، وإلّا فإن الطبقات الدنيا ستتعرض للتلف"، كما يقول رباي.

وعليه، فإن التصاميم المنضّدة (layered) الثلاثية الأبعاد تتطلب في الوقت الحالي على الأقل بنياناً مختلفاً. "إن تحقيق توصيلات أعلى كثافةً يقود فجأةً إلى أن الفصل التقليدي بين المعالج والذاكرة يغدو غير منطقي. إذ عليك إعادة النظر في طريقة إجراء الحوسبة" كما يقول رباي. فالاقتراح ليس مقترحاً مجرداً تماماً. " والمزايا التي تتمتع بها بعض التطبيقات– خاصةً تعلُّمَ الآلة والتعلُّمَ العميق، اللذين يتطلبان تكاملاً مكثفاً للذاكرة والمنطق– تذهب أدراج الرياح." أضف إلى التحدّيات أن التصميم الثلاثي الأبعاد يزيد من احتمال حدوث أعطالٍ داخل الجذاذة. ويضيف رباي " إن انتاج جذاذة تعمل عملاً سليماً 100% أمرٌ مستحيل. ويجب أن تكون المنظومة متسامحةً مع الأعطال[2] (fail-tolerant) وأن تتعامل مع الأخطاء."

بقطع النظر عن المنهجية ومجموعة التقانات، فإن الباحثين تُركوا في نهاية المطاف دون خيارٍ مثالي. من دون اختراقٍ جذري، وعليهم إعادة التفكير بالطريقة الأساسية لإنجاز الحوسبة والمعالجة.

يقول كونت إن هناك إمكانَيْن خارج إطار توجهات التقانة الحالية.

يتطلب أحد هذين الإمكانَيْن تغيراتٍ جذرية، ولكن تُحَدُّ هذه التغيرات بتلك التي نصادفها "تحت الستار" في البنيان المِكروي. وهذا ما حصل إلى حدٍّ ما في عام 1995، ما عدا أنه " يجب علينا اليوم استعمال منهجيات أكثر جذريةً" ، كما يقول رباي. فمثلاً، طُرِحَ استعمال المواد الفائقة الناقلية التي تعمل عند درجات حرارة فائقة الانخفاض، في المُخدِّمات وفي

[2] الخاصية التي تُمكِّن المنظومة من الاستمرار بالعمل على نحوٍ سليم في حال تعطّل بعض مكوّناتها. (المراجع)

الحوسبة العالية الأداء، كأحد الحلول الممكنة. في الوقت الحاضر، تستثمر منظمة "فعالية المشاريع البحثية الذكية المتقدمة الأمريكية" (Intelligence Advanced Research Projects Activity, IARPA) استثماراً كبيراً في هذه المنهجية في برنامجها Cryogenic Computing Complexity (C3). تُصنع هذه البوابات المنطقية غير التقليدية على نطاقٍ ضيق، وهي أكبر بنحو 200 مرة من الترانزستورات الحالية.

أمّا الإمكان الآخر، كما يقول كونت فهو " أن نتحمّل الألم ونغيّر نموذج البرمجة." وبالرغم من تقديم العديد من الأفكار والمفاهيم، فإن معظمها يُركِّز على إيجاد مسرِّعات ذات دوالّ ثابتة (غير قابلة للبرمجة) للأجزاء الحرجة من البرامج الهامة. " تكمن الميزة هنا في أنه عندما تنزع قابلية البرمجة، فإنك تزيل كل الطاقة المستهلكة في تعليمات الجلب (fetching) وفك الترميز (decoding)." هناك إمكان آخر– وهو الإمكان الذي يأخذ شكله فعلياً– يقوم على إبعاد الحوسبة عن وحدة المعالجة المركزية (CPU)، وتوجيهها نحو المعطيات الفعلية. بشكلٍ أساسي، يمكن للبنيانات المتمحورة حول الذاكرة (memory-centric)، وهي في مرحلة التطوير في المخابر، أن تقوّي المعالجةَ دون أي تحسيناتٍ في الجذاذات.

ختاماً، يتحرى الباحثون طرقاً مختلفةً تماماً في الحساب، ويشمل ذلك النماذج العصبونية المورفية (neuromorphic) والكمومية، التي تعتمد على طرائق مختلفة عن فون–نُوْيْمان[3] مستوحاة من الدماغ والحوسبة الكمومية. يقول رباي إن المعالجات تتوجه بالفعل في هذا الاتجاه. ومع ظهور التعلُّم العميق والحوسبة الإدراكية (cognitive)، يزداد استعمال أكداس وحدات المعالجة البيانية (graphics processing unit, GPU) لتسريع الأداء بذات التكلفة من الطاقة أو أقل من وحدات المعالجة المركزية التقليدية. وبالمثل، فإن جذاذات الهواتف النقالة وإنترنت الأشياء أتت بمتطلبات معالجة مختلفة تماماً. "في بعض الحالات، يُغيّر هذا نموذج خفض متطلبات المعالجة للمنظومة، ولكن مع وجود تجهيزات (devices) في كل مكان. قد نرى بلايين أو تريليونات التجهيزات التي تُكامل الحوسبة والاتصالات مع الاستشعار والتحليل ومهام أخرى.

في الحقيقة، مع تطور المعالجة المرئية وتحليل المعطيات الكبيرة وعلم التشفير (cryptography) والحقيقة المزيدة والحقيقة الافتراضية (Augmented Reality/Virtual Reality, AR/VR) وتقانات متقدمة أخرى، من المرجح أن يزاوج الباحثون بين منهجيات مختلفة لإنتاج جذاذات جاهزة (boutique chips) تلائم تجهيزةً أو حالةً خاصة أفضل ملاءمةً. ويخلص كونت إلى القول "يتجذر المستقبل بالتعددية وبناء تجهيزات لِسَدِّ حاجات بنيانات الحاسوب الواعدة أكثر من غيرها."

قراءات إضافية

- *Conte, T.M., and Gargini, P.A.*, On the Foundation of the New Computing Industry Beyond 2020, Executive Summary, *IEEE Rebooting Computing and ITRS*. September 2015. http://rebootingcomputing.ieee.org/images/files/pdf/prelim-ieee-rc-itrs.pdf

- *Lam., C.H*., Neuromorphic Semiconductor Memory, 3D Systems Integration Conference (3DIC), 2015 International, 31 Aug.-2 Sept. 2015. http://ieeexplore.ieee.org/document/7334566/

- *Claeys, C., Chiappe, D., Collaert, N., Mitard, J., Radu, I., Rooyackers, R., Simoen, E., Vandooren, A., Veloso, A., Waldron, NH. Witters, L., and Thean, A.* Advanced Semiconductor Devices for Future CMOS Technologies, *ECS Transactions*, 66 (5) 49-60 (2015) 10.1149/06605.0049ecst

[3] جون فون–نُوْيْمان (Von-Neumann): رياضي وفيزيائي أمريكي من مواليد هنغاريا 1903–1957 م، وهو غير جون هنري نيومان (Newman): كاردينال في الكنيسة البريطانية 1801–1890 م. (المدقق اللغوي)

©The Electrochemical Society. 2015. https://www.researchgate.net/ profile/C_Claeys/publication/277896307_Invited_Advanced_Semiconductor_ Devices_for_ Future_CMOS_Technologies/ links/565ad44408aefe619b240bcc.pdf

- *Cheong, H.* Management of Technology Strategies Required for Major Semiconductor Manufacturer to Survive in Future Market, Graduate School of Management of Technology, Hoseo University, Asan 336- 795, Korea, Procedia Computer Science 91 (2016) 1116 – 1118. Information Technology and Quantitative Management (ITQM 2016). http://www. sciencedirect.com/science/ article/pii/S1877050916313564

الحوسبة اللامتجانسة: وجدت لتبقى

HETEROGENEOUS COMPUTING: HERE TO STAY*

محمد زهران
ترجمة: د. محمد سعيد دسوقي
مراجعة: د. ندى غنيم

ازداد ذكر عبارة *الحوسبة اللامتجانسة* في السنوات القليلة الماضية، وسيستمر ذكرها لسنوات قادمة، لأن الحوسبة اللامتجانسة وُجدت لتبقى. ما هي الحوسبة اللامتجانسة ولماذا ستصبح المعيار؟ كيف نتعامل معها، من الجانبين البرمجي والعتادي؟ يوفِّر هذا المقال إجابات لبعض هذه الأسئلة كما يطرح وجهات نظر مختلفة حول بعضها الآخر.

لنبدأ بالأسئلة السهلة، ما هي الحوسبة اللامتجانسة؟ بالمختصر، هي أسلوب يكون فيه للعقد الحاسوبية المختلفة إمكانات مختلفة و/أو طرائق مختلفة لتنفيذ التعليمات. لذلك، فالنظام اللامتجانس هو نظام متوازٍ (نظم النواة الوحيدة أصبحت من التاريخ القديم). عندما ظهرت النظم المتعددة النوى، كانت متجانسة – أي إن جميع النوى كانت متشابهة. كان الانتقال من البرمجة التسلسلية إلى البرمجة المتوازية، التي عادةً ما كانت مجالاً محصوراً بالمبرمجين المختصين، قفزةً كبيرة. في الحوسبة اللامتجانسة، تكون النوى مختلفة.

يمكن أن يكون للنوى المقدرات البنيانية نفسها – على سبيل المثال، تَوَفُر قدرة النيسبة الفائقة (hyperthreading capacity) (أو عدمه)، عرض السُلَّمية الفائقة (superscalar width) نفسه، حساب الأشعة، وما إلى ذلك. إلا أنه، حتى النوى المتشابهة في هذه القدرات، يكون لديها نوع ما من اللاتجانس، وذلك لأن كل نواة لها الآن تدرج ديناميكي للفلطية والتردد خاص بها (DVFS (dynamic voltage and frequency scaling. فالنواة التي تقوم بعمل أكثر سترتفع حرارتها ومن ثَم ستقلل ترددها وتصبح بذلك أبطأ. لذلك، حتى النوى التي لها المواصفات نفسها يمكن أن تكون لامتجانسة. هذا هو النوع الأول من اللاتجانس.

يتعلق النوع الثاني بالنوى التي لها مقدرات بنيانية مختلفة. من الأمثلة على ذلك، معالجٌ ذو نوى بسيطة متعددة (مثلاً، مسألة واحدة (single issue)، ليست خارج الترتيب (out-of-order)، دون تنفيذ تحزّري (speculative execution)، مع بعض النوى السمينة (مثلاً، مع تقنية النيسبية الفائقة، ونوى سلّمية فائقة مع تنفيذ خارج الترتيب وتنفيذ تحزّري).

**نُشِر هذا البحث في مجلة Communications of the ACM، المجلد 60، العدد 3، آذار (مارس) 2017، الصفحات 42 – 45.*

يتعلق هذان النوعان الأوليان من اللاتجانس بالنوى التي لها نموذج تنفيذ البرمجة التسلسلية نفسه – أي إنَّ كل نواة تبدو كأنها تنفذ التعليمات بالتسلسل حتى إن كان هنالك تحت الغطاء ضمنياً نوع من التوازي بين التعليمات. يمكنك مع هذه الآلة المتعددة النوى كتابة رماز متوازٍ، لكن كل نيسب (أو إجرائية) تنفذه النواة بأسلوب يبدو تسلسلياً. ماذا لو كانت هنالك عقد حاسوبية لا تعمل بهذه الطريقة؟ هذا هو النوع الثالث من اللاتجانس.

في هذا النوع من اللاتجانس يكون للعقد الحاسوبية نماذج تنفيذ مختلفة. توجد حالياً عدة أنواع مختلفة من العقد، وأكثرها شهرةً وحدة المعالجة البيانية GPU (graphic processing unit)، المستعملة حالياً في تطبيقات مختلفة عديدة إلى جانب البيانيات. على سبيل المثال، تُستعمل وحدات المعالجة البيانية بكثرة في التعلم العميق، وخصوصاً في جزء التدريب. وتُستعمل أيضاً في تطبيقات علمية متعددة وتعطي أداءً برتبة أفضل من أداء النوى التقليدية. إن سبب تعزيز الأداء هذا هو أنَّ وحدة المعالجة البيانية تستعمل نموذج تنفيذ التعليمة الواحدة (أو النيسب) والمعطيات المتعددة. لنفترض أن لديك مصفوفة كبيرة وتحتاج إلى ضرب كل عنصر في هذه المصفوفة بثابت. باستعمال نواة تقليدية، يجري عمل ذلك على عنصر واحد في كل مرة، أو في أفضل الحالات، على عدة عناصر في كل مرة. باستعمال وحدة معالجة بيانية، يمكنك ضرب جميع العناصر مرة واحدة، أو إجراء عدد قليل من التكرارات إذا كانت المصفوفة كبيرة جداً. تتفوق وحدة المعالجة البيانية في العمليات المتشابهة المستقلة على حجوم كبيرة من المعطيات.

يعدُّ نموذج صفيفة بوابات قابلة للبرمجة حقلياً FPGA (field-programmable gate array) نموذجاً آخر للحوسبة يختلف عن الأسلوب التسلسلي التقليدي. نعلم جميعاً أن البرمجيات والعتادات متكافئة منطقياً، أي إن ما تستطيع فعله بالبرمجيات يمكنك أيضاً فعله بالعتاديات. تكون الحلول العتادية أسرع بكثير لكنها غير مرنة. يحاول نموذج FPGA سدّ هذه الثغرة، فهو دارة يمكن للمبرمج تشكيلها لتنجيز دالة ما. افترض أنك تحتاج إلى حساب تابع كثير الحدود على مجموعة من العناصر. تجري ترجمة تابعٍ كثير الحدود واحدٍ إلى عشرات من تعليمات لغة المجمّع. يعتبر نموذج FPGA خياراً جيداً إذا كان عدد العناصر الضرورية لحساب الدالة ليس كبيراً إلى حدٍ يتطلب استعمال وحدة معالجة بيانية، وليس صغيراً بحيث يُنْجَزَ بفعالية ضمن نواة تقليدية. استعملت نماذج FPGA في كثيرٍ من الحشود العالية الأداء. مع استحواذ شركة إنتل Intel في السنة الماضية على شركة ألتيرا Altera، أحد اللاعبين الكبار في سوق FPGA، من المتوقع حصول تكامل أكبر بين نماذج FPGA والنوى التقليدية. بدأت ميكروسوفت أيضاً باستعمال نماذج FPGA في مركز معطياتها (مشروع كاتابيلت Catapult).

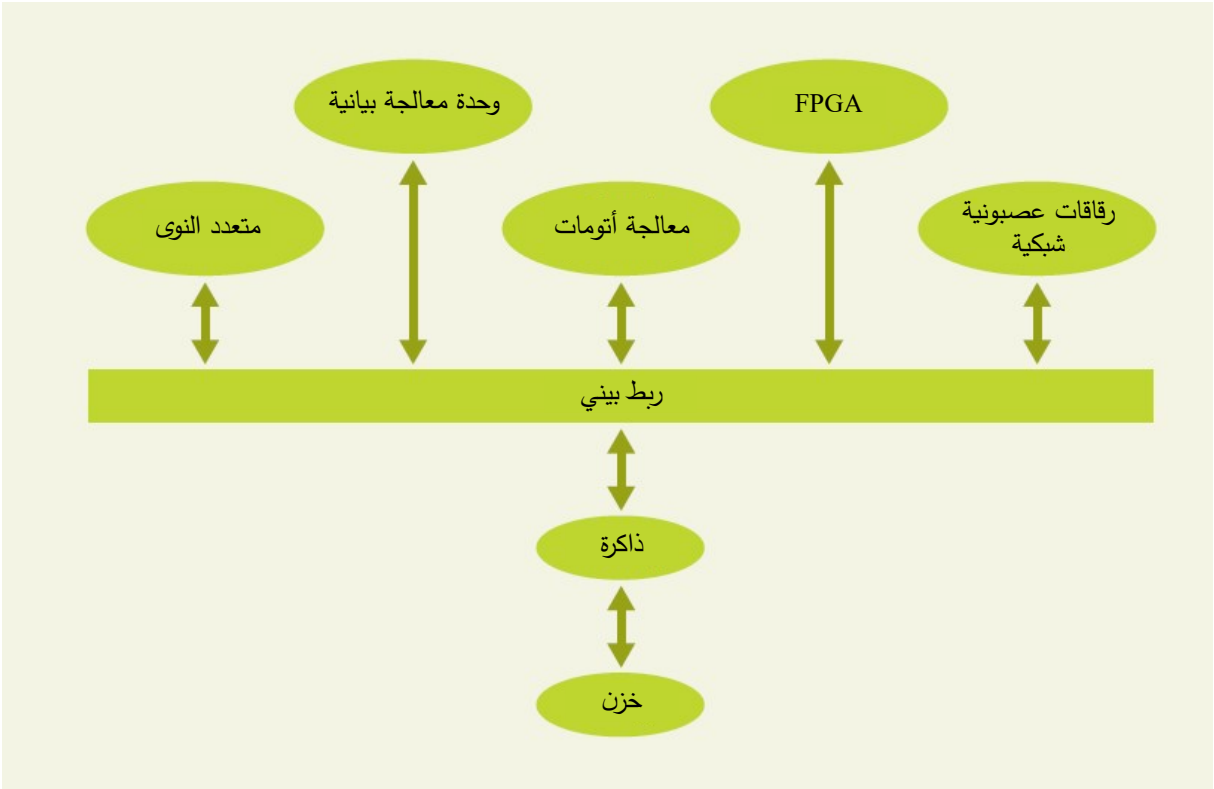
قامت شركة ميكرون Micron [3] حديثاً بإضافة عضو جديد إلى خيارات عقدة الحوسبة هو معالج الأتومات AP (automata processor). يناسب هذا المعالج تحليل البيانات (graph analysis)، ومطابقة الأشكال (pattern matching)، وتحليل المعطيات (data analytics)، وعلم الإحصاء. فكِّر به باعتباره مسرعاً للتعابير النظامية يعمل على التوازي. إذا كان بإمكانك صياغة المسألة التي بين يديك كتعبير نظامي، فيمكنك أن تتوقع الحصول على أداء أعلى من ذلك الذي توفره وحدات المعالجة البيانية. بني AP باستعمال نماذج FPGA لكنه صُمِّمَ ليكون أشد فاعليةً في معالجة التعابير النظامية.

إلى جانب عقد الحوسبة المذكورة سابقاً، هنالك عقد معالجة متعددة أخرى مثل معالج الإشارة الرقمية DSP (digital signal processor) والدارة المتكاملة المحددة بحسب التطبيق ASIC (application-specific integrated circuit).

تستهدف هذه العقد شرائح صغيرة من التطبيقات، وليست متعددة الاستعمالات كما هو حال العقد المشار إليها سابقاً. تبدأ الرقاقات العصبونية الشبكية المستلهمة من الدماغ، مثل رقاقة TrueNorth الخاصة بشركة IBM عصراً من الحوسبة الإدراكية [2]. حالياً، تُستعمل الحوسبة الإدراكية، التي تتزعمها رقائق Watson وTrueNorth الخاصة بشركة IBM، بعد الأداء المذهل لنظام الذكاء الصنعي الحاسوبي Watson في "Jeopardy"، في التطبيقات الطبية، كما يجري استكشاف مجالات أخرى. مع ذلك، من المبكر قليلاً مقارنتها بالنوى الأخرى ذات الأغراض العامة.

لا يهتم باقي المقال إلاّ بالنوى التقليدية (مع إمكانات مختلفة) GPU، وFPGA، وAP. يظهر الشكل المرافق الصورة الكبيرة لنظام حوسبة لامتجانسة، مع ذلك، بسبب تكلفته البرمجية، من غير المحتمل إيجاد نظام بمستوى اللاتجانس الموجود في الشكل. سيتضمن النظام الحقيقي مجموعة جزئية من هذه الأنواع فقط.

ما فائدة هذا التنوع من عقد الحوسبة؟ تكمن الإجابة في الأداء وفي فعالية الطاقة. افترض أنَّ لديك برنامج بنياسب صغيرة متعددة. الخيار الأفضل في هذه الحالة هو مجموعة من النوى الصغيرة. إذا كان لديك عدد قليل جداً من النياسب المعقدة (على سبيل المثال، بيانات تحكم بالتدفق معقدة مع ملاحقة المؤشرات)، فعند ذلك ستكون النوى المعقدة (مثل النوى السمينة الفائقة التوسع) هي ما يجب اختياره. إذا أسندت النياسب المعقدة إلى نوى بسيطة، فالنتيجة ستكون أداءً ضعيفاً. وإذا أسندت النياسب البسيطة إلى النوى المعقدة، فستستهلك طاقة أكبر مما تحتاج. لوحدات المعالجة البيانية فعالية أداء/طاقة جيدة للتطبيقات ذات المعطيات المتوازية. ما نحتاجه هو آلة عامة الأغراض يمكنها تنفيذ برامج ذات نكهات



نظام لامتجانس عمومي.

مختلفة مع فعالية عالية من حيث الأداء والطاقة. الطريقة الوحيدة لعمل ذلك هي الحصول على آلة لامتجانسة [3]. حالياً، لمعظم التجهيزات، من الحواسيب المحمولة إلى الحواسيب اللوحية إلى الهواتف الذكية، بنيانات لامتجانسة (عدة نوى ووحدة معالجة بيانية)، ومن المتوقع ازدياد اللاتجانس في المستقبل القريب (جداً). كيف يجب أن نتعامل مع هذا التحول النوعي من التجانس إلى اللاتجانس؟

تحديات العتاديات

توجد عدة تحديات على مستوى العتاديات. أولها هرمية الذاكرة. نظام الذاكرة هو أحد نقاط اختناق الأداء في أي نظام حاسوبي. برغم اتباع المُعالِجات لقانون مور حتى سنوات قريبة ماضية، مما صنع قفزات جيدة في الأداء، فإنَّ نظم الذاكرة لم تتبع ذلك القانون. لذلك، هنالك ثغرة أداء كبيرة بين سرعة المعالج وسرعة الذاكرة. وُجِدَت هذه المشكلة منذ عصر النواة الوحيدة، وما يجعلها أكثر تحدياً في هذه الحالة هو هرمية الذاكرة التشاركية (عدة مستويات من الذاكرة الخبيئة يليها الذاكرة الرئيسية). ما الذي يتشارك في كل مستوى من الخبيئات؟ تستهدف كل من النوى الحسابية التي ناقشناها برنامجاً (أو نيسباً أو إجرائية) له مميزات تختلف عن تلك التي تستهدفها النوى الحسابية الأخرى. على سبيل المثال، تتطلب وحدة المعالجة البيانية عرض حزمة أكبر، في حين تتطلب نواة تقليدية نفاذاً أسرع. بالنتيجة، ما نحتاجه هو هرمية ذاكرة تقلل من التداخل بين النوى المختلفة، لكنها تتعامل بفعالية مع المتطلبات المختلفة لكلٍ منها.

إنَّ تصميم هرمية كهذه ليس أمراً سهلاً، خصوصاً لأنَّ نظام الذاكرة إضافةً إلى إشكاليات الأداء هو مصدر غير بديهي لاستهلاك الطاقة. هذا التحدي هو موضوع بحث مكثف في القطاعين الصناعي والأكاديمي. إضافةً إلى ذلك، فإننا نقترب من عصر الذاكرة اللامتلاشية (nonvolatile memory). كيف يمكن استعمالها أفضل استعمال؟ لاحظ هنا اللاتجانس في وحدات الذاكرة: للذاكرة الخبيئة (SRAM)، وللذاكرة المتلاشية (DRAM)، وللذاكرة اللامتلاشية (MRAM, STT-RAM, PCM, ReRAM) وتقانات كثيرة متعددة.

التحدي الآخر على مستوى العتاد هو الترابط البيني: كيف يجدر بنا أن نربط النوى المختلفة ووحدات هرمية الذاكرة؟ تستهلك الأسلاك الغليظة طاقة أقل، لكنها تنتج عرض حزمة أقل لأنها تاخذ مساحة أكبر على الرقاقة. هنالك حجمٌ متنامٍ من البحث في الترابط البيني الضوئي. الطبولوجيا (حلقة، كعكة مستديرة، تشابك)، والمادة (نحاس، ضوء)، والتحكم (شبكة على رقاقة، بروتوكولات)، هي موضوعات بحث ساخنة على مستوى الرقاقة، وعلى مستوى اللوحة، وفيما بين اللوحات.

مازال هنالك تحدٍ آخر يتمثل في توزيع حمل التشغيل بين النوى المختلفة للحصول على أفضل أداء مع أقل استهلاك للطاقة. يجب أن نجد إجابة هذا السؤال بالبحث الكامل في كدسة الحوسبة كاملاً، من الخوارزميات إلى تقانة المعالج.

يعني الانتقال من لوحة وحيدة إلى لوحات متعددة وإلى حواسيب عالية الأداء أيضاً انتقالاً من الذاكرة المشتركة إلى الذاكرة الموزعة. هذا ما يجعل الترابط البيني وتوزيع حمل التشغيل أكثر تحدياً.

تحديات البرمجيات

تُعدُّ الحالة تحدياً كبيراً أيضاً على مستوى البرمجيات. كيف سنبرمج هذه الوحوش؟ البرمجة التسلسلية صعبة. البرمجة المتوازية أصعب. تواجه البرمجة المتوازية للآلات اللامتجانسة تحدياً شديداً إذا كنا نهتم بالأداء وفعالية الطاقة. هنالك اعتبارات عديدة: كم من العتاد يجب أن نكشف للمبرمج، قياسات النجاح، والحاجة لنموذج برمجة جديد (أو لغة جديدة).

قبل محاولة الإجابة عن هذه الأسئلة، نحتاج إلى مناقشة المسألة الأبدية لإنتاجية المبرمج مقابل أداء البرمجيات المولَّدة. كانت الحكمة الشائعة هي أنّهَ لزيادة الإنتاجية يجب إخفاء العديد من جوانب العتاديات عن المبرمج. تجعلك الكتابة بلغة بايثون (Python) أكثر إنتاجيةً من الكتابة بلغة C، التي هي أكثر إنتاجيةً من الكتابة بلغة المجمِّع (assembly). أليس كذلك؟ ليست الإجابة بهذه السهولة، لأن العديد من مساقات بايثون، على سبيل المثال، ليست إلا تغليفات لمساقات بلغة C. مع تكاثر الآلات اللامتجانسة، سينشئ مبرمجو الأداء (performance programmers) المزيد من المكتبات ليستعملها مبرمجو الإنتاجية (productivity programmers). مع ذلك، حتى مبرمجو الإنتاجية يحتاجون لاتخاذ بعض القرارات الصعبة: كيف يجب تقسيم التطبيق إلى نياسب (أو إجرائيات) مناسبة للعتاديات المتوفرة (يمكن أن يتطلب ذلك تجربة خوارزميات مختلفة)، وما هي أجزاء البرنامج التي لا تتطلب أداءً عالياً ويمكن تنفيذها ضمن نمط أقل استهلاكاً للطاقة (الأجزاء التي تتطلب عمليات إدخال وإخراج مثلاً)؟

يطرح تعريف قياسات النجاح عدداً من التحديات لمبرمجي الإنتاجية والأداء معاً. ما هي قياسات النجاح لبرنامج كُتب لأجل آلة لامتجانسة؟ للعديد من قياسات النجاح هذه مميزات مشتركة مع تلك الخاصة بالرماز المتوازي التقليدي في الآلات المتجانسة. القياس الأول، طبعاً، هو الأداء. كم هي الزيادة في السرعة التي حصلت عليها نسبةً إلى الإصدار التسلسلي ونسبةً إلى الإصدار المتوازي في الحوسبة المتجانسة؟

القياس الثاني هو قابلية التوسع. هل يتوسع برنامجك عند إضافة نوى أكثر؟ إنَّ قابلية التوسع في الحوسبة اللامتجانسة أكثر تعقيداً منها في الحالة المتجانسة. في الأخيرة، تضيف فقط عدداً أكبر من النوى نفسها. في الآلات اللامتجانسة، لديك خيارات أكثر: إضافة نوى أكثر من نوع ما، أو إضافة وحدات معالجة بيانية أكثر، أو ربما دارات FPGA. كيف سيتصرف البرنامج في كل حالة؟

القياس الثالث للنجاح هو الموثوقية. كلما أصبحت الترانزيستورات أصغر أصبحت أكثر عرضةً للأعطال، سواءً العارضة منها أو الدائمة. هل تترك أمر التعامل مع الأعطال للعتاديات، أم لبرمجيات النظام، أم أن المبرمج لديه ما يقوله؟ لكل استراتيجية محاسنها ومساوئها. من جهة، إذا تُرك ذلك للعتاديات أو لبرمجيات النظام، سيصبح المبرمج أكثر إنتاجية. من جهة أخرى، لدى المبرمج معلومات أفضل من النظام ليقرر كيفية تحقيق هبوط مقبول في الأداء إذا نقص عدد النوى نتيجة الفشل، أو إذا أنتجَ نيسبٌ ما نتيجةً خاطئةً بسبب عطل عارض. يمكن أن يكون لدى المبرمج، مثلاً، إصداران من المساق الجزئي نفسه: أحدهما للتنفيذ على وحدة معالجة بيانية والآخر على نوى تقليدية متعددة.

المحمولية أمر آخر. إذا كنت تكتب برنامجاً مخصصاً لآلة محددة جيداً، فإنَّ القياسات الثلاثة الأولى كافية، لكن إذا كنت تكتب برنامجاً للاستعمال العام على آلات حوسبة لامتجانسة مختلفة، فعندها يجب أن تضمن المحمولية. ماذا

يحدث مثلاً لو جرى تنفيذ برنامجك على آلة تتضمن FPGA بدلاً من وحدة معالجة بيانية GPU؟ لن يكون هذا المشهد مستبعداً في المستقبل القريب.

الاستراتيجية الفضلى

بأخذ هذه الأسئلة والاعتبارات، ما هي الاستراتيجية الفضلى؟ هل يجب أن نُدخل نماذج برمجة (ولغات) جديدة، أم يجب أن نصلح/نحدِّث النماذج الموجودة؟ لدى علم النفس ما يقوله. كلما كان لدى الشخص خيارات أكثر كان ذلك أفضل –حتى الوصول إلى عتبة ما. بل أبعد من ذلك، سيصبح الناس محملين فوق طاقتهم وسيلزمون اللغة التي يستعملونها. لكننا يجب أن نكون حذرين عند *إصلاح* لغةٍ ما. كانت لغة بيرل Perl تسمى "لغة الكتابة فقط" "write-only language". لا نريد أن نقع في الفخ نفسه، فتحديد أي لغة يجب إصلاحها/تعديلها قرار بالغ الصعوبة، وأي قرار خاطئ سيكون له تكلفة كبيرة. في الحوسبة اللامتجانسة، تبدو لغة الحوسبة المفتوحة OpenCL (Open Computing Language) مرشحةً جيدةً لآلات الذاكرة المشتركة، لكنها يجب أن تكون أكثر سهولة في الاستعمال. ماذا بشأن الذاكرة الموزعة؟ هل واجهة تمرير الرسائل MPI (Message Passing Interface) جيدة إلى حدٍ كافٍ؟ هل تعتبر أيٌّ من اللغات/النماذج المتوفرة حالياً الموثوقةَ قياساً للنجاح؟

يبدو أنَّ الأسلوب الأفضل مؤلف من شقين: نماذج جديدة مخترعة ومختبرة في القطاع الأكاديمي، في حين يحدث الترشيح في القطاع الصناعي. كيف يحدث الترشيح؟ يحدث عندما تحصل نقطة انعطاف في عالم الحوسبة. من الأمثلة على نقط انعطاف سابقة: الانتقال من النواة الوحيدة إلى النوى المتعددة وظهور وحدات المعالجة البيانية. نشهد حالياً زوجاً من نقاط الانعطاف في الوقت نفسه: القرب من الحوسبة برتبة إكسا عملية بالثانية (exascale computing) وظهور إنترنت الأشياء. تُعدُّ الحوسبة اللامتجانسة التقانةَ المؤهلة لكليهما.

الحوسبة اللامتجانسة موجودة الآن وستبقى. يتطلب استعمالها بالصورة الفضلى إعادةَ النظر في كدسة الحوسبة كاملةً. على مستوى الخوارزميات، تذكَّر أنَّ الحساب الآن أقل تكلفة من النفاذ إلى الذاكرة ونقل المعطيات. تحتاج نماذج البرمجة للتعامل مع مسألة الإنتاجية مقابل الأداء. تحتاج المترجمات لتعلم استعمال العقد اللامتجانسة. مازال أمام المترجمات طريق طويل لتقطعه لأنها لم تنضج بعد في مجال الحوسبة المتوازية عموماً كنضجها في مجال الحوسبة التسلسلية. يجب أن تتعلم نظم التشغيل خدعاً جديدة. يجب أن يقرر مهندسو بناء الحواسيب ما هي العقد التي يجب وضعها معاً للحصول على أشد الآلات فعاليةً، وكيفية تصميم هرمية الذاكرة، والطريقة الفضلى لربط كل هذه الوحدات معاً. على مستوى الدارات وعلى مستوى تقانة المعالج، لدينا قائمة أمنيات طويلة من الموثوقية والطاقة والتوافق والتكلفة. هنالك ثمرات معلقة على جميع مستويات كدسة الحوسبة، وكلها جاهزة للقطاف إذا كنا نستطيع اكتشاف الأشواك.

المقالات ذات الصلة

- Computing without Processors, *Satnam Singh*, http://queue.acm.org/detail.cfm?id=2000516
- FPGA Programming for the Masses, *David F. Bacon, Rodric Rabbah, and Sunil Shukla*, http://queue.acm.org/detail.cfm?id=2443836
- A Conversation with John Hennessy and David Patterson, http://queue.acm.org/detail.cfm ?id=1189286

المراجع

[1] HAS Foundation: http://www.hsafoundation.com/.
[2] IBM Research. The cognitive era: https://www.research.ibm.com/cognitive-computing/.
[3] Micron Automata processor: http://www.micronautomata.com/.

المؤلف

محمد زهران (Mohamed Zahran) أستاذ مساعد في علم الحاسوب في جامعة نيويورك. تغطي اهتماماته البحثية عدة مجالات في بنيان الحاسوب وتفاعل العتاد والبرمجيات.

تعزيز الإبداع بالحوسبة

FOSTERING CREATIVITY THROUGH COMPUTING*

Aman Yadav, Steve Cooper
ترجمة: م. وسام شريفة
مراجعة: د. سعيد دسوقي

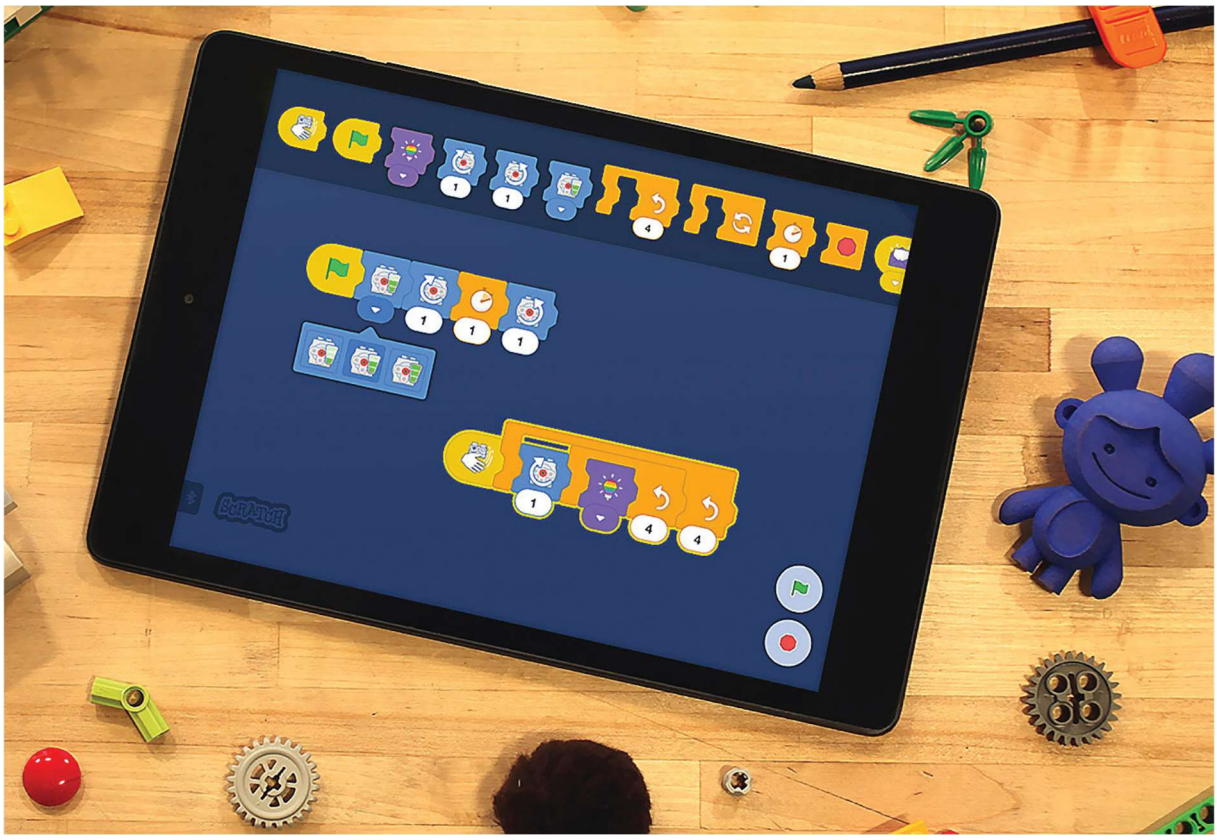
كيف يمكن الاستفادة من أدوات التفكير الإبداعي والحوسبة في دعم المساعي البشرية للإبداع.

للحوسبة القدرة على إتاحة الفرص للمستعملين لتوسيع تعبيرهم الإبداعي لحل المشاكل، وخلق مصنوعات حسابية، وابتكار معارف جديدة. تغير أيضاً الطبيعة الانتشارية للحوسبة وامكان الوصول للأدوات الرقمية التعليم في السنوات الاثني عشر الأولى (K-12) إذ ينتقل الطلاب من كونهم مجرد مستهلكين للمحتوى إلى منخرطين بالموضوع وذلك ببنائهم مصنوعات حسابية. لنأخذ لغة "السكراتش" على سبيل المثال، فهي واحدة من العديد من الأدوات المصممة لتعليم الأطفال البرمجة، وتأتي مع عدد من مستويات الدعم للمعلمين لينفذوها في إعدادات التعليم الرسمي وغير الرسمي. تقدم "السكراتش" فرصة للطلاب للتعبير عن إبداعهم عن طريق القصص، اللُعب، والرسوم المتحركة. ومع أن "السكراتش" يمكن أن تكون أداة فعالة فإنها غالباً تستعمل بوصفها أكثر بقليل من أداة العرض في الصف المدرسي. أظهرت الدراسات على مستعملي لغة "السكراتش" أن عدداً قليلاً من المشاريع يستعمل المتغيرات أو بِنى معطيات تدفق التحكم. وفي الوقت الذي تقدم فيه بيئة "السكراتش" أداة سهلة وقوية ("low floor high ceiling") للمبتدئين للغوص في البيئة بدون إحباط، فإن العديد من الطلاب لا يتقدمون إلى مستوى أعلى. تُمكّن الأدوات مثل "السكراتش" الطلاب من إظهار إبداعهم كما لم يسبق لهم من قبل. على كل حال إن لطريقة تعليم هذه الأدوات من قبل المدرسين واستعمالها من قبل الطلاب الأثر الكبير في استمرارية الطلاب في الإبداع. وعلى حين تستعمل "السكراتش" على نطاق واسع، نعلم القليل فقط عن مدى تأثيرها في تفكير الطلاب الإبداعي.

تعليم علوم الحاسوب والإبداع فيها في السنوات الدراسية الـ 12 الأولى (K-12)

انتقد الاستاذ "كين روبنسون Ken Robinson" بشدة في حديثه في برنامج TED المشهور، المؤسسات التعليمية، مدعيا أننا "نقوم بتعليم الناس بعيداً عن إبداعهم". ربما بسبب هذا أو ربما فقط بسبب الإقرار بأهمية تطوير إبداع الطلاب خلال تعليمهم علوم الحاسوب، صنفت كلية بورد College Board الإبداع كواحدة من الأفكار السبع العظيمة، كجزء من دورة مبادئ علوم الحاسوب المتقدمة التنسيب (AP CSP). في الحقيقة، وردت كلمة "إبداع" بأشكالها المختلفة 62 مرة في توصيف الدورة

**نُشِر هذا البحث في مجلة Communications of the ACM، المجلد 60، العدد 2، شباط (فبراير) 2017، الصفحات 31 – 33.*



تنفيذ نموج اولي من كتل "السكراتش"، من مدونة مطوري غوغل، بالتعاون مع فريق مخبر الوسائط المتعددة للسكراتش في معهد ماساتشوستس للتكنولوجيا MIT.

AP CSP وامتحاناتها. والحجة في ذلك هي أن الحوسبة تعزز الإبداع بالسماح للأفراد بالانتقال من مجرد مستعملين للتكنولوجيا إلى بُناة أدوات يمكن أن يكون أثرها كبيراً في المجتمع. تبين دورة مبادئ علوم الحاسوب (CSP) كيف أنه باستطاعة الحوسبة تمكين الناس ليس فقط من استعمال الحوسبة للتعبير الإبداعي، ولكن أيضاً "لتوسيع الأشكال التقليدية للتعابير والتجارب البشرية [1]". إن الفهم واستعمال الحوسبة (كأدوات البرمجة والخدمات)، والمعرفة العميقة في اختصاص ما، والتعبير الإبداعي يسمح للأفراد بخلق مصنوعات حسابية أو/و حل المشاكل. تُحسِّن العلاقة التي يقيمها الفرد مع أدوات الحوسبة ليس فقط من تعبيره الإبداعي، بل يمكنها أيضا أن تؤدي إلى أشكال جديدة من المصنوعات [3]. على سبيل المثال استعمل الموسيقي "إنيس إكزانكس (Iannis Xenakis) "التوزيعات الاحتمالية في بدايات الخمسينيات من القرن الماضي لتركيب الموسيقا التي سماها موسيقا إحصائية. ولكي يُسرِّع الحسابات الإحصائية بدأ" إكزانكس" بالبرمجة. لا يقوم برنامجه فقط على حساب تركيبات الأوركسترا (النسب المئوية للمقاطع) ولكن أيضا على إسناد علامة موسيقية لآلة معينة. سمحت المعرفة العميقة بالاختصاص(الموسيقا) وفهم "إكزنكس" لبرمجة الحاسوب بدمج قوة الحوسبة في تركيب موسيقا عشوائية (stochastic).

سلط إطار عمل مبادئ علوم الحاسوب الضوء على الجانب الإبداعي لما استطاع إكزانكس تحقيقه مع الحوسبة – توسيع الاشكال التقليدية للخبرة والتعبير البشريين. على أية حال تجدر الإشارة إلى أنه بالرغم من الإلحاح على الإبداع في إطار عمل مبادئ علوم الحاسوب، فإنه الفكرة الوحيدة من الأفكار السبع الكبيرة التي تُختبر صراحةً في امتحان دورة مبادئ

علوم الحاسوب الجديدة. وفي حين نتفق على كون الإبداع بناء معقد للفهم والتقييم، نستطيع التعلم من مجالات أخرى مثل علم النفس حيث يوجد عدد من مقاييس الإبداع (فمثلاً، امتحان تورانس للتفكير الإبداعي (Torrance Test of Creative Thinking)، اختبار استعمال بديل جليفورد (Guilford's Alternate Uses Test)، اختبار والاس وغوكان (Wallace and Kogan's test) وغيرهم). ربما يقوم مجلس الكلية (College Board) بوضعِ نموذجٍ واضحٍ لوضعِ درجاتٍ للإبداع في امتحان مبادئ علوم الحاسوب (CSP)، كوسيلة لقياس الإبداع، ضمن ملفات الطلاب المُسلَّمة.

يبقى السؤال هنا عن كيفية عرض هاتين القوتين البارزتين للإبداع والحوسبة، ضمن سياق لتخصصات معينة، على الطلاب، وهذا ما يؤدي إلى حل المسائل المهيكلة بشكل سيِّئ في القرن الواحد والعشرين. كيف نستعمل الحوسبة في مختلف المجالات التي تواجه الطلاب في طيف تعليمهم الدراسي الابتدائي والثانوي؟ سنركز في ما تبقى من هذا العمود على كيفية تدريس الحوسبة بطريقة تحسِّن من التفكير الإبداعي للطلبة. من المهم أن نلاحظ أننا لسنا جدداً في اقتراح كيفية تطوير الإبداع لدى الطلاب. منذ قرن تقريباً، بحث جاكس هادامارد (Jacques Hadamard) في الاختراع في الرياضيات (كمثال على الاختراع عموماً) لكي يفهم الإجرائية التي يستعملها الرياضيون العظماء للاختراع [2]. يقدم كتاب" The Cambridge Handbook of creativity" نظرة شاملة إلى أبحاث الإبداع، من علاقتها بالإدراك إلى خصوصية مجالاتها إلى تقييم الإبداع. وفي حين كان يجري عمل عظيم لفهم وتنمية عملية الإبداع، أثار اهتمامنا بوجه خاص بعمل الباحثين في الإبداع كروبرت ومايكل روت–بيرنستين (Robert and Michele Root-Bernstein)، وسوف نناقش في هذا العمود كيف أن عملهم على التفكير الإبداعي يمكن أن يغني تعليم علوم الحاسوب.

التفكير الإبداعي بواسطة علوم الحاسوب

يتضمن الإبداع مجموعة من أدوات التفكير التي تتداخل مع أساسيات علوم الحاسوب، والتي تستطيع بدورها دعم تطوير التفكير الإبداعي. نوجز هنا بعض أدوات التفكير الإبداعي وتداخلاتها مع علم الحاسوب، ونناقش كيفية استعمال الحوسبة في دعم المساعي البشرية الإبداعية.

تعد *المراقبة* واحدة من أهم أدوات التفكير الإبداعي التي تذهب بعيداً لتشمل السمع والشم والتذوق، والتي تسمح لنا مجتمعة بالحصول على المعرفة [4]. أداة أخرى مرتبطة بالإبداع هي التصور، الذي يسمح للأفراد بمعاينة /تصور مظهر الأشياء غير الموجودة مادياً. مثلا استعمل الفيزيائي ريتشارد فايمان (Richard Feynman) الصور المرئية كحل لمسألةٍ ما قبل القفز إلى المعادلات الرياضية لإيجاد الحل. تعد القدرة على الإدراك عن طريق الملاحظة والتصور أمراً بالغ الأهمية للتفكير الإبداعي والابتكار. في علوم الحاسوب، تؤدي الملاحظة دوراً ملحوظاً للباحثين في مجال الرؤية الحاسوبية "الرؤية الصنعية"، مثل الرسوم الحاسوبية والرؤية الحاسوبية. فمثلا مجموعة "معاينة رؤية الرسوم" (Graphics Vision Visualization) في جامعة ترينتي (http://gv2.cs.tcd.ie) تعتمد بقدر كبير على كيفية فهم البشر لتطوير المعاينات مثل الوكلاء الافتراضين. يسمح النفاذ إلى هذه الأدوات للطلاب بالتفكير بطريقة أقوى وساعدهم على إيقاظ مهارات التفكير الإبداعي لديهم المتعلقة بالإدراك. نستطيع تحسين مهارات الفهم للطلاب وتوسيع مفهوم التخيل بجعلهم يتخيلون كيف يُبنى الرماز المصدري والخوارزميات وكيف تعمل، هذه القفزة ذهابا وإيابا بين شرح الخوارزمية وكتابة الرماز بحد ذاته.

فن الفيل: فريد ولكن غير إبداعي.

تعد القدرة على *التجريد* –تخفيض المعلومات والتفاصيل بغية التركيز على المفاهيم المتعلقة بحل المشكلة– أداة أخرى من أدوات التفكير الإبداعي الأساسية ويُسلط الضوء على أهميتها في علوم الحاسوب بكونها واحدة من الأفكار الكبرى في دورة مبادئ علوم الحاسوب.

فيما يخص لعلماء الحاسوب، فإن التجريد مفهوم أساسي. في ورشة عملنا الخاصة بلغة أليس (Alice) مع الأساتذة، تُحَفِّز الحاجة إلى التفكيك واستعمال الطرائق (كشكل من أشكال التلخيص) عن طريق استنباط حل لمسألة ما دون استعمال التجريد. في حالتنا هذه نعلّم التّنين الطيران باستعمال الحركات البدائية، والدوران، وطرائق اللف على كل من الأجزاء المناسبة من جسم التنين. من الصعب فهم الرماز الذي لم يُقَسَّم إلى عدد من الأجزاء او حتى تعديله في وقت لاحق. يسمح التجزيء باستعمال الطرائق للمبرمج بالتفكير بالطريقة بمستوىً أعلى من التجريد. بمجرد حصول المبرمج على تفاصيل الطريقة للعمل، لا يعود من الضروري التفكير في التعليمات التفصيلية التي تشكل الطريقة. تعمل الطريقة ببساطة كما يفترض بها ان تعمل. لاحظنا أن نهجنا مشابه قليلًا للطريقة التي اتبعها مهران سهامي (Mehran Sahami) في مقرر منهجية البرمجة iTunes University، مع كاريل (karel) وهي مهمة ربوطية (robot task). تدرس الوسطاء بطريقة مشابهة. في مثالنا في

أليس (Alice) بدلاً من خلق طريقة وحيدة للتنين للطيران إلى الفارس، وطريقة أخرى لطيران التنين إلى الملك، من الممكن وضع موسطات لهدف رحلة طيران التنين.

أخيرا، يُعدُّ التشكيل أداة أخرى من أدوات التفكير المركزية للإبداع، التي تحتوي على كلا الأمرين: المقدرة على تعُّرف الأشكال إضافة إلى المقدرة على تشكيلها [4]. يؤدي تعرُّف الأشكال دوراً مهماً لعلماء الحاسوب الذين يعملون في مجال تعليم الآلة، خاصة عندما يعود الأمر لاستخلاص المعلومات الصحيحة المعتمدة على تمييز الأشكال من مجموعات بيانات كبيرة. وبالمشابهة، يُعدُّ تكوين الأشكال المفتاح لوضع جميع أجزاء المعلومات في تصور علمي صحيح. يستطيع الطلاب على سبيل المثال استعمال أدوات الحوسبة للتعلم عن تشكيل وتمييز الأشكال في المعطيات عن طريق جداول البيانات في الصفوف الابتدائية بداية، ثُم باستعمال توابع إحصائية ورياضية بلغة بايثون (Python) في الصفوف الثانوية.

الخلاصة

طرحنا في هذا العمود كيف أن الحوسبة تزود الطلاب بآلية قوية لتدعم تفكيرهم الإبداعي. ومع ذلك، نحن بحاجة للنظر بعناية إلى كيفية استعمالنا لإمكانات الأدوات الحاسوبية بدلاً من مجرد وضعها أمام الأستاذة وطلابهم. وأيضا نحن بحاجة إلى معالجة كيفية استعمال الأدوات الرقمية من قبل الأستاذة وطلابهم للحصول على ادوات التفكير الإبداعي كما ناقشنا في هذا العمود. نحن أيضا بحاجة لتطوير مقاييس تسمح لنا بتقييم طريقة وأداء الحوسبة وما إذا كانت تدعم الاجرائيات التي تساعد على إبداع الطلاب. ننتهي مع الأمل، نأمل أن يؤدي الاستعمال المخطط جيداً لأدوات الحوسبة إلى نتائج إبداعية أكثر من تلك الناتجة عن إعطاء ريشة رسم ومجموعة ألوان لفيلٍ، كما يظهر في الصورة المرفقة، فمع أن فن الفيل فريد، فإنه لا يُظهر–على الأقل من وجهة نظرنا– أي درجة من الإبداع من قبل الفيل.

المراجع

[1] AP computer science principles draft curriculum framework: 2014; https://advancesinap.collegeboard.org/stem/computer-science-principles.

[2] Hadamard, J. *The Psychology of Invention in the Mathematical Field.* Dover Publications, 1954.

[3] Mishra, P. and Yadav, A. Of art and algorithm: Rethinking technology and creativity in the 21st century. *TechTrends 57*, 3 (2013), 10–14.

[4] Root-Bernstein, R.S. and Root-Bernstein, M.M. *Sparks of Genius.* Houghton Mifflin, Boston, 1999.

المؤلفان

أمان ياداف (ayadav@msu.edu): استاذ مساعد في كلية تكنلوجيا التعليم في جامعة ولاية مشنغن.

ستيف كوبر (scooper@unt.edu): استاذ مساعد في قسم علوم الحاسوب في جامعة نيبراكاسا.

البرمجة القابلة للتعلُّم: الكتل وما بعدها

LEARNABLE PROGRAMMING: BLOCKS AND BEYOND*

David Bau, Jeff Gray, Caitlin Kelleher, Josh Sheldon, Franklyn Turbak
ترجمة: د. نزار الحافظ
مراجعة: أ. مروان البواب

إطارات عمل جديدة للكتل تفتح الأبواب لتجربة أكبر للمبتدئين والمحترفين على حد سواء .

أدى التدافع العالمي لتوسيع المشاركة في علوم الحاسوب إلى تعاظم الاهتمام بالبرمجة المعتمدة على الكتل. وغدت الكتل المرئية مستعملة في كثير من أدوات البرمجة (انظر الشريط الجانبي: أدوات تعليمية معتمدة على الكتل). وبدأ ملايينُ الطلاب يستكشفون البرمجة بواسطة هذه الأدوات في الدورات التدريبية والأنشطة، مثل: ساعةِ ترميز (Hour of Code) في موقعِ Code.org؛ فالكتل تتيح للمبتدئين إنشاء البرامج دون أن يعانوا من إحباطاتِ التركيب النحْوي (syntax) لهذه البرامج (الشكل 1).

وثمة اهتمام متزايد في تطويرِ لغاتِ الكتل ودراستها؛ ففي مؤتمرِ VL/HCC (Visual Languages and Human-Centric Computing) الذي عقد عام 2015، توسَّعت حلقةُ عملٍ صغيرة بعنوان (الكتل وما بعدها)[1] لتصبح حدثًا كبيرًا، وقد تضمنت 51 مشاركة و 36 مقدِّمًا. وكانت مشاركة الباحثين فيها تدور حول: لغاتٍ جديدة للكتل، وابتكاراتٍ خاصة بالواجهات، وتطبيقاتِ الكتل الخاصة بمجالات محددة، وطُرقٍ لجعل لغات الكتل أشدّ فاعليةً ونفاذًا إليها من طرَف المبرمجين المتنوعين.

توضِّح هذه المقالة تأثير الكتل في قابلية تعلُّم البرمجة؛ فتبدأ بمراجعة الدراسات المتعلقة بفعالية

أفكار رئيسية

- برزت بيئاتُ برمجةِ الكتل بصفتها طريقةً شائعةً للتعريف بالترميز وخطوةً نحو اللغات التقليدية المعتمدة على النصوص، ولكن يمكن استعمالها أيضًا لكتابة "رماز حقيقي".
- تُحسِّن بيئاتُ الكتل قابليةَ التعلُّم للمبتدئين عن طريق: تفضيل التمييز (recognition) على التذكّر (recall)؛ والتقليل من الحِمل المعرفي (cognitive load) وذلك بتقطيع النماذج الحسابية النمطية إلى كتل؛ وتسخير التعامل المباشر بالكتل لمنع الأخطاء وتعزيزِ فهْم بِنية البرنامج.
- تُحسَّن قابليةُ التعلُّم أيضًا بواسطة الميْزات الرئيسية التي تتجاوز الكتل؛ ومنها: البيئات الموصولة بالخط (online)، والتجريدات العالية المستوى، والحالة المرئية، والأمثلة التي يسهل العثور عليها.
- الأدوات متاحةٌ لتمكين الأشخاص من تحسينِ لغة الكتل الخاصة بهم.

**نُشِر هذا البحث في مجلة Communications of the ACM، المجلد 60، العدد 6، حزيران (يونية) 2017، الصفحات 72 – 80.*

[1] http://cs.wellesley.edu/blocks-and-beyond

لغات الكتل، ثم تناقش الميزاتِ الرئيسية للغات الكتل وارتباطها بالتعلم. وأخيرًا، تُلقي نظرة على تطبيقات الكتل في المجالات

الشكل 1. حزمة Snap! هي مثال على بيئةِ برمجةٍ معتمدة على الكتل. يسحب المبرمجون الكتل من لوحة عناصر البرمجة (يسارًا) إلى مساحة العمل (في الوسط)، حيث يمكن تجميع الكتل في برامج. يوفِّر Snap! أيضًا نافذةَ إخراج (أعلى اليمين) ومنتقي أشباح (أسفل اليمين).

الجديدة وتناقش الأدوات التي تمكِّن من إنشاءِ لغةِ كتل خاصة بالمبرمج.

إن مشاهدة المبرمجين المبتدئين وهم ينشئون برامجهم الأولى باستعمال الكتل هو مصدر إلهامٍ وقلقٍ في آنٍ معًا، إذ إن تَمكُّنهم من استعمال الكتل سيسرِّع في بناء إبداعات معقدة، غالبًا ما تكون مبهجة. لكنهم، وبسرعةٍ مماثلة، سيملؤون شاشتهم برمازٍ غير متقَن ومعقد.[22] وقد يَخشى المبرمجُ المخضرم، الذي يعاين تجميعةَ (assembly) المبتدئ غير المنظمة، ألَّا يكون تقميش كتلٍ مفعمة بالألوان ذا علاقة بـ "الرماز الحقيقي". ولكن ما هو "الرماز الحقيقي"، ولماذا نتعلمه؟

ما هو "الرماز الحقيقي؟" إن الغرض من الأداة المعتمدة على الكتل هو تسهيل تعلُّم البرمجة. لكنْ يمكن أن يكون لتعليم البرمجة نهايتان متمايزتان: بناء الخبرة لدعم البرمجة الاحترافية، والقدرة على تحقيق الأهداف الأخرى عن طريق إنشاء البرامج.

إن هاتين النهايتين ليستا متماثلتين بالضرورة. فقد لاحظ مصممو لغة سكراتش (Scratch) أن المبرمجين "الذين يرون البرمجة وسيلةً للتعبير، لا مسارًا نحو مهنة ما، يجدون أن سكراتش كافية لاحتياجاتهم."[28] جرى تصميمُ بيئةِ الكتل في لغةِ

GP[2] لتمكين "المبرمجين العاديين" من إنشاء برامج أكثر تعقيدًا من أيّ وقت مضى، وفي الوقت ذاته إزالةِ القيود التي من شأنها أن تجبرهم على الابتعاد عن الكتل. [24] وقبل مناقشةِ آثارِ تعلُّم البرمجة المعتمدة على الكتل، لا بد من التنبيه على أن من قصر النظر أن نفترض أن مبرمجي الغد سيبرمجون باستعمال اللغات والنظم الحالية؛ إذ إن كل جيل من المبرمجين يغيّر ثقافة الترميز، وسيستمر تعريف "الرماز الحقيقي" في التطور.

ومع ذلك، تظل مسائل التعلُّم الأساسية دون تغيير. فمثلًا، يمكننا أن نسأل الطلاب الذين يواصلون دراسة البرمجة التقليدية: ما مدى الفائدة من إدخال البرمجة المعتمدة على الكتل؟ لقد اختُبرتْ هذه المسألة مباشرة في قاعات الدراسة.

قياس نقل التعلُّم: تشير الأبحاث إلى أن تعلُّم لغة الكتل يمكن أن يحسِّن التعلُّمَ اللاحقَ للّغة النصية التقليدية؛ فقد بيّنتْ دراسةٌ أجريتْ على تلاميذ الصف العاشر الذين يتعلمون لغة C# أو جافا (Java)، [1] أن التلاميذ الذين تلقوا دورةً في لغة سكراتش في الصف التاسع تعلموا هاتين اللغتين بوتيرةٍ أسرع، وفهموا الحلقات (loops) البرمجية فهمًا أفضل، وكانوا أكثر انخراطًا وثقةً من أقرانهم الذين لم يتلقوا سكراتش. لكنْ، في الاختبار النهائي، لوحظ وجود اختلاف كبير في محورٍ واحدٍ فقط من ثلاثة محاور معرفية. وفي دراسة أجريتْ في مدرستين ثانويتين، [25] أتمَّ التلاميذُ الذين لديهم خبرةٌ قليلة أو معدومة في البرمجة وإعدادٌ ضعيف في الرياضيات، صفَّ البرمجة CS0 باستعمال لغة أليسْ (Alice) قبل بدء دورة Java CS1؛ فسُجّل أولًا تحسُّنٌ في درجات الطلاب في أليسْ (بمتوسط (GPA[3]) قدره 3.0 مقابل 1.2 للطلاب الذين لم يتلقوا لغة أليسْ)، ثم ازديادُ النسبة المئوية للطلاب الذين اتبعوا دورات إضافية في علوم الحاسوب (CS) (88٪ مقارنة بـ 47٪).

جرى اختبار مسألةِ كونِ هذه الآثار ناتجةً عن منح الطلاب المزيد من التحضير في صفٍّ إضافي بإنشاءِ دوراتٍ تَجمع بين مقدمةٍ عن لغة معتمدة على الكتل وانتقالٍ إلى لغةٍ تقليدية؛ فأشارت التقارير من الدورات التي تُدرَّس فيها سكراتش قبل جافا أو C إلى تحسُّنِ مشاركةِ الطلاب وفهمِهم لبعض المفاهيم .[39،19] وفي إحدى الدراسات التي ركزت على نقل التعلُّم، [8] جرى تعديلُ دورةٍ تمهيديةٍ في جافا في جامعة CMU بحيث تبدأ بلغة أليسْ، فتحسَّن متوسط أداء الطلاب في هذا الصف الذين استعملوا كلتا اللغتين بنسبة 10٪ أو أكثر في كل قسم من أقسام اختبار جافا النهائي ذاته؛ ومن ذلك اختبار: تقييم التعابير، وبِنى التحكم، والصفيفات (arrays)، والتعامل مع تعريفات الصفوف.

هذه النتيجة رائعة، لأن المرء يمكن أن يَفترض أن إنفاق المزيد من الوقت على البرمجة بالكتل يعني إنفاق وقت أقل لتعلُّم لغة جافا. وقد استَعملتِ الدراسةُ إصدارًا من أليسْ ولَّد رمازَ جافا من كتلِ أليسْ، وأصلًا تربويًّا توفيقيًّا رَبط ربطًا صريحًا بين مفاهيم البرمجة في أليسْ وجافا.

ويُذكر أن دراسات أخرى لدوراتِ CS1 تحولت من الكتل إلى النص لم تتضمن هذه الميزات، واجهت تحدياتٍ محتملةً للتعلُّم باستعمال الكتل؛ [10، 27] منها أنّ التحوُّل من لغة الكتل إلى النص يمكن أن يشمل كلًّا من التغيير في التركيب النحوي والدلالي (syntax & semantics)، وهنا يقترح شابيرو (Shapiro) وأهرنز (Ahrens) أنْ يعلَّم هذان التحوُّلان على

---

[2] GP هي لغة برمجة يجري بناؤها حاليًّا. الغرض منها تقديمُ برمجةٍ أكثر تطورًا ذات تصميمٍ شبيهٍ بسكراتش للمبرمجين. تعني GP: "extensible portable General purpose block language for casual Programmers" لغةَ الكتل المتعددة الأغراض القابلة للحَمْل والتوسيع للمبرمجين العاديين. ومن الغايات المستقبلية للغة GP أن تعمل بسرعةٍ وعلى أي منصة. (المصدر: https://en.scratch-wiki.info/wiki/GP_(programming_language)). (المترجم)

[3] Grade Point Average: متوسط الدرجات، هو مقياس يلخص الإنجاز الأكاديمي للتلميذ أو الطالب. (المترجم)

انفراد وذلك بتقديم التركيب النحوي على التعميم الدلالي. [32] وثَمة حاجة إلى إجراء بحث إضافي لتعرُّف الظروف التي تكون فيها الكتل فعالة.

يشار إلى أن العديد من الدورات التمهيدية في علوم الحاسوب تستعمل – في هذه الأيام – نهج الكتل قبل النص؛ ففي دورةِ CS50 من جامعة هارفارد، ينتقل الطلاب من سكراتش إلى C؛ وتتقدم دورةُ CS10 في جامعة بيركلي من لغة Snap! إلى لغة بايثن (Python)؛ وتَستعمل دورةُ مبادئِ علوم الحاسوب (CSP) التابعةُ لمؤسسةِ Project Lead The Way (PLTW) كلًّا من لغة سكراتش وأداة App Inventor قبل الانتقال إلى لغة بايثن؛ وتنتقل دورةُ CSP App Lab التابعة لمؤسسة Code.org من كتلِ دروبلت (Droplet) إلى جافا سكريبت (JavaScript).

لِمَ كانت الكتل قابلة للتعلُّم؟

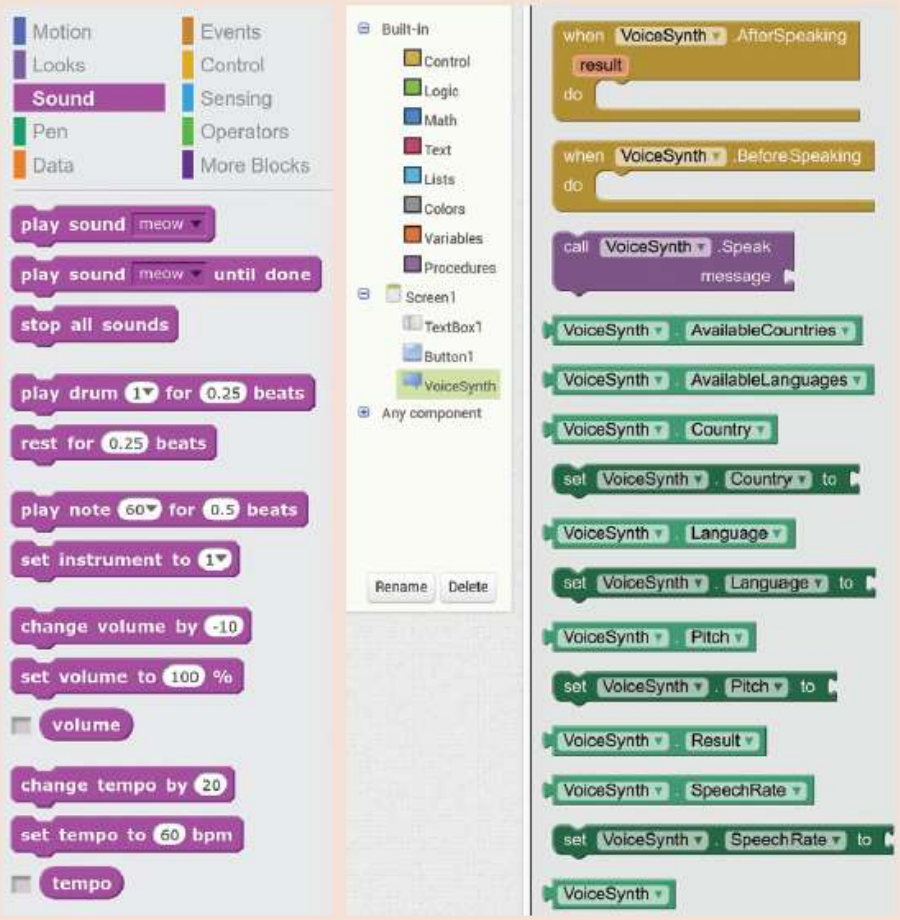
في عام 2004، حدَّد Ko, Myers و Aung [15] ستةَ عوائقَ تتعلق بالتعلُّم واجهها غير المبرمجين في مهام البرمجة. ثلاثة منها – *الانتقاء، والاستعمال، والتنسيق* – هي صدى لصعوبةِ تجميع بسيط لبرنامجٍ ما. ونحن نعتقد أن قابلية تعلُّم لغاتِ الكتل تنشأ من كيفيةِ معالجةِ تحدياتِ قابليةِ الاستعمال المتعلقة بعوائق التعلُّم الثلاثة الآتية:

1. إنَّ تعلُّم مفردات البرمجة أمر صعب. أما الكتل فإنها تبسِّط هذه المشكلة، وذلك لأن التقاط كتلة من لوحة الكتل أسهل بكثير من تذكّر كلمة: فالكتل تعتمد على *التمييز بدلًا من التذكّر*.
2. الرماز صعب الاستعمال لأن فيه حِملًا معرفيًّا عاليًا عند المبرمجين الجدد. أما الكتلُ فإنها تقلِّل هذا الحِمل عن طريق تقطيع الرماز (*chunking code*) إلى عددٍ أصغر من العناصر ذات المعنى.
3. إن تجميع الرماز عرضة للخطأ. أما الكتل فإنها تساعِد المبرمجين على تجميع الرماز دون الوقوع في أخطاء أساسية وذلك بتوفير تعامل مباشر مقيَّد للبِنية (فمثلًا، لا يوجد بين مفهومَي كتلتين غير متوافقين أية أجزاء تَربط بينهما).

التمييز مقابل التذكر. تتضمن البرمجة بلغةٍ بسيطة (أو مكتبتها) عادةً مفرداتٍ من نحوِ 100 إلى 200 كلمة؛ فمثلًا، تحوي لغةُ HTML 100 أمارة (tag) و 100 خاصية (attribute)، وتحوي SQL نحو 200 كلمة رئيسية (keyword)، وتحوي لغة Scratch مثلهما 130 كتلة. وإن تذكّر 100–200 مفهوم قد يؤدي إلى إرباك المبرمج الجديد.

أما لغات الكتل – وخلافًا للغات النصية – فإنها تَرتبط ارتباطًا وثيقًا ببيئات البرمجة الخاصة بها، وتعتمد جميع بيئات الكتل تقريبًا على عدد قليل من اصطلاحات الواجهة التي تعالج المشكلات الرئيسية المتعلقة بقابلية الاستعمال. وأحد هذه الاصطلاحات التعاملُ مع المفردات عن طريق تنظيم الكتل في لوحات مترابطة وظيفيًّا على الشاشة.

تختلف لوحات الكتل (palettes) عن قوائم الإكمال التلقائي (autocomplete menus) لمحررات الرماز الاحترافية، في أنها تبقى ظاهرة ولا تختفي، وفي أنها تنظِّم المفاهيم بحسب الموضوع بدلًا من الاسم. إن هذا التصميمُ يبسِّط الاكتشافَ والاستكشاف معًا. يوضح الشكل 2–أ لوحةَ كتل الصوت في بيئة سكراتش، وهي مرجِع توضيحي يُظهِر جميع الطرق الثلاثة عشر للصوت في تلك البيئة. وثمة تنظيم مماثل في بيئات أخرى ذات كتل أكبر. وللمساعدة على التغلب على التعقيدِ المتعلق بإنشاءِ تطبيقاتِ الجوال، توفر بيئةُ App Inventor مجموعةً ديناميكية من الكتل، وتتيح كتلًا إضافية في برامجَ تتفاعل مع مزيد من المكونات (الشكل 2–ب). وماتزال العبارة الاصطلاحية الأساسية حتى الآن هي نفسها في سكراتش؛ أي: لوحات قابلة للاستكشاف منظمة بحسب الوظيفة.

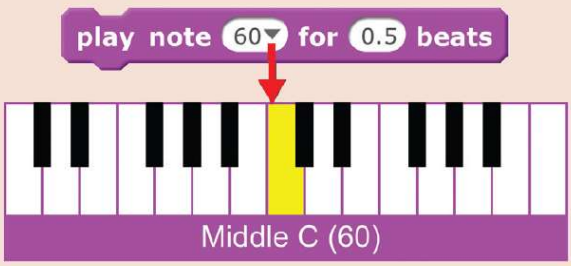


الشكل 2. لوحة الصوت في سكراتش (أ) ولوحة تركيب الصوت فيApp Inventor (ب). تيسِّر اللوحات اختيارَ عناصر البرمجة بالاستفادة من سهولة *التمييز* مقابل *التذكّر*. تنظِّم اللوحاتُ المفاهيمَ حسب الموضوع، لا حسب الاسم، وتظل مفتوحةً عند استعمالها، لِتسمح للمبرمج باكتشاف الكتل والتعامل معها اعتمادًا على وظيفتها.

إنّ تذكُّر ترتيبِ المؤثِّرِ فيه (operand) ونوعِه وقيمتِه الصالحة أمر شاق أيضًا للمبرمجين الجدد. أما لغات الكتل، فإنها تعالج العديد من هذه الصعوبات بتوفير الكتل مع قيم مغتفَلة (default) للمؤثَر فيه، وقوائم منسدلةٍ ومحرِّراتٍ متخصصة لتحديد المؤثِّرات فيها، وكلماتٍ إضافية للدلالة على معاني المؤثَّر فيه (الشكل 3–أ).

تقطيع المعلومات في بيئة الكتل. تُبدي لغاتُ البرمجة حِملًا معرفيًّا عاليًا للطالب الذي يتعلم تركيبًا نحْويًّا (syntax) جديدًا. على سبيل المثال، تأمّل حلقة `for` في التركيب النحْوي للغة جافا سكريبت:

```
for (var i = 0; i < 50; i++) {_ _ _}
```




```
playNoteFor(60, 0.5)
```

Scratch (أ) Python (ب)

الشكل 3. تُظهِر الكتلُ البِنيةَ شكلًا مرئيًّا (أ. سكراتش) بدلًا من استعمالِ علامات الترقيم، وبذلك فهي تساعد على قابلية التعلم باستعمال لغةٍ واضحة، وقيمٍ مغتفَلة، ومنتقيات قِيَم.

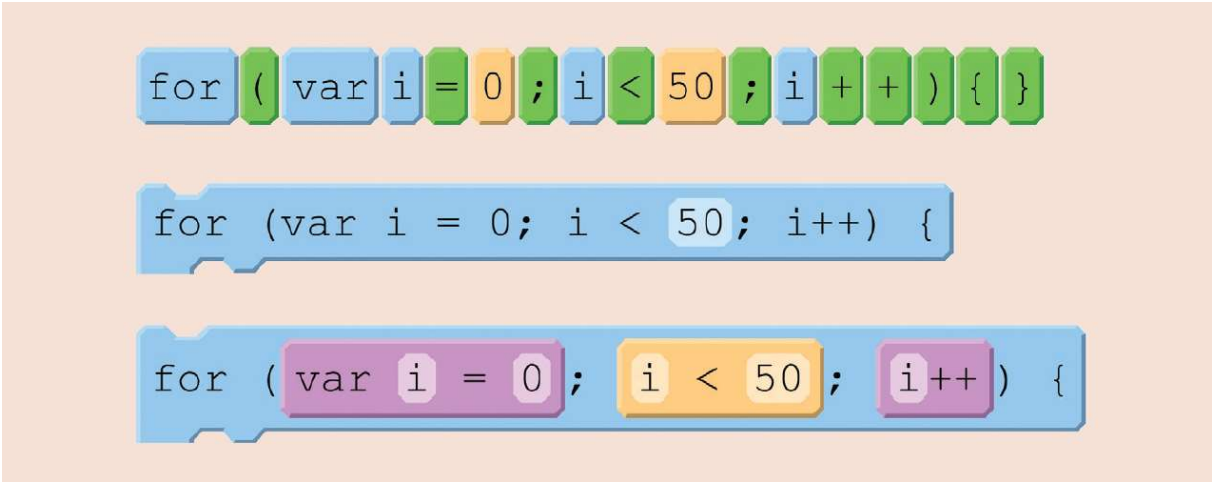
إن هذا التدوين الكثيف يعدُّ عائقًا للمبتدئين. وحسب تعبير أحد الطلاب: "ثَمة إرباك حقيقي يخصّ جافا سكريبت في فهم الأقواس المختلفة "( )" و "{ }" وكل ما كان على شاكلتها". [38]

ولتوضيح هذه الصعوبة، افترضْ أن هذا الرماز يحوي: خمس كلمات [for var i i i]، و 10 علامات ترقيم [{}(=;<;++)]، ورقمين [0 و 50]، أيْ ما مجموعه 17 وحدة معلومات. أَثبتت دراساتُ قدراتِ الإدراك البشري أن لدى الناس ذاكرة عمل تتسع لسبعِ قِطَعٍ من المعلومات. [23] ولهذا فإن محاولةَ فهمِ سطر الرماز هذا الذي فيه 17 بندًا من المعلومات تمكن أن تُربك ذاكرةَ العمل لدى المبرمِج الجديد.

وواقع الحال أنه لا توجد مشكلة لدى ذوي الخبرة من مبرمجي جافا سكريبت في فهم سطر الرماز المشار إليه، وذلك لأنهم تعلموا تفسير الرماز في قِطَعٍ أكبر منه. ولما كانت حلقة for تتبع نمطًا شائعًا جدًّا، فإن قراءتها يمكن أن تكون في قطعتين: الأولى، حلقة `for` المعتادة التي تَستعمل نمط التكرار التقليدي (`i` يبدأ من 0 ويزداد بمقدار 1)؛ والثانية، اختيار القيمة 50 لتكون الحد الأقصى للحلقة. يوضح (الشكل 4) طرقًا مختلفة لتقطيع الرماز.

أما الكتل، فإنها تساعد على تخفيف الحِمل المعرفي بإرشادِ المبرمجين الجدد إلى كيفيةِ قراءةِ قِطَعٍ أكبر. ففي دورة مبادئ علوم الحاسوب التابعة لموقع Code.org، تُرسِم الكتل لتمثيل الحلقات (`for`) في جافا سكريبت تمامًا كما يَرى الخبيرُ الرماز؛ إذ تُستعمل قطعتان، ولكلّ كتلةٌ منفردة ومَقبس منفرد للقيمة العظمى للحلقة. يمكن أيضًا تبسيط التعقيد بوضعِ قطعٍ داخل قطع. على سبيل المثال، يَكشف Code.org النقاب عن بِنية `for` ذات حبيبيّةٍ أدقّ (finer-grained) في أجزاء أكثر تقدمًا في السياق نفسه، كما هو موضح في الجزء السفلي من (الشكل 4).

يحسِّن استعمالُ الكتل لتقطيعِ الرمازِ المقروئيةَ (قابليةَ القراءة) (readability) حتى في حالة الأوامر البسيطة، وذلك لأن الكتل يمكن أن تتخلى عن علامات الترقيم التي يستعملها الرماز النصي لتعيين البِنية، وتَستعملُ كلماتٍ تفسيرية بدلًا من ذلك. على سبيل المثال، وكما هو موضح في (الشكل 3–ب)، يتطلب استدعاءٌ بسيطٌ في لغة بايثن قراءةَ الفواصل (delimiters) ومعرفةَ ترتيبِ المحدِّدات (arguments)، على حين أن الكتلة المكافئة لهذا الاستدعاء تُقرأ طبيعيًّا في سكراتش، باستعمال التجريدات المناسبة (على سبيل المثال، لوحة مفاتيح البيانو).



الشكل 4. ثلاثُ طرقٍ لقراءةِ حلقةِ `for` مقطَّعة: الأولى قراءةٌ بسيطةٌ للرماز (أعلى) تفسِّر رمازَ الحلقة بـ 17 قطعة، والثانية قراءةٌ خبيرةٌ للرماز (وسط) تفسِّر أكثرَ أشكال الحلقة شيوعًا بقطعة منفردة، تضاف إليها قطعةٌ ثانية لحدِّ الحلقة 50، والثالثة قراءةٌ بديلة (أسفل) تفسِّر ثلاثَ فقراتٍ بثلاث قِطَع. يَستعمل موقعُ Code.org التفسيرَ الأوسطَ لتعليم طلاب المدارس الثانوية الحلقات بادئ الأمر، ويجري التحول إلى التفسير السفلي عندما يصبح الطلاب على درايةٍ بحلقات `for`.

وهكذا تساعدُ الكتلُ – بالاستعانة بتنظيم الرماز على شكلِ قِطَعٍ مرئية – المبرمجينَ الجدد على التركيز على ما يعنيه الرماز بدلًا من التدوين المستعمَل لكتابته.

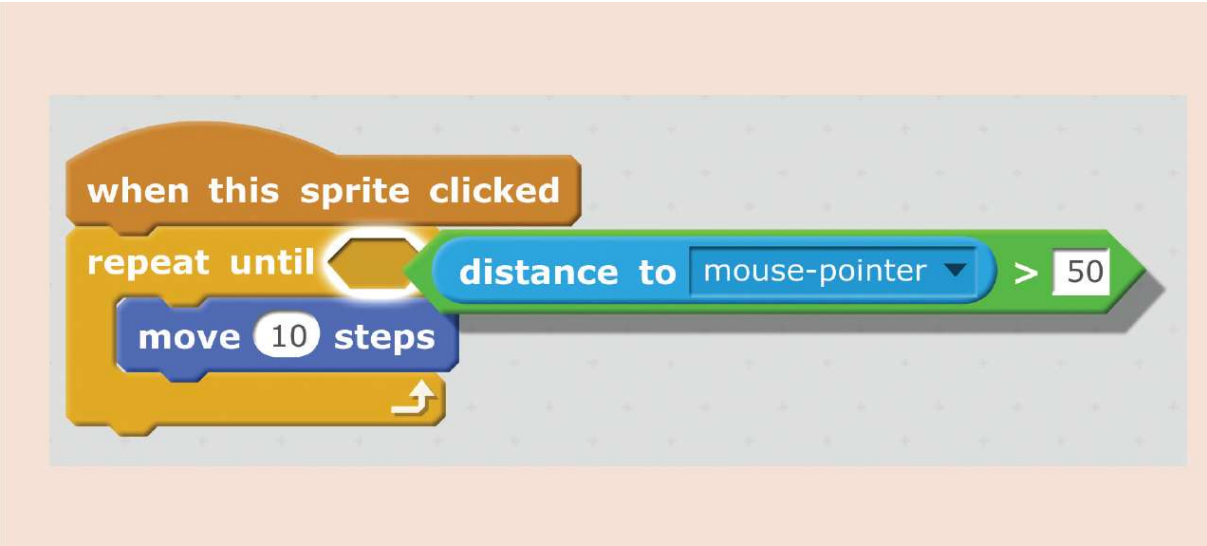
التعامل المباشر مع البِنية المرئية. يخفِّف الشكلُ المرئي للكتل من عبء تجميع الوحدات الصحيحة نحْويًّا، وذلك بالكتابة حرفًا حرفًا. وثَمة مزايا أخرى للتعامل المباشر مع قطعِ البرنامج التي تحوي قيودًا مرئية.

من هذه المزايا أن الكتل يمكن أن تساعد على منع الأخطاء عن طريق جعل قواعد البرنامج مرئية؛ إذ يمكن رؤية الكتل شكلًا من أشكال التحرير (editing) الموجَّه بالتركيبِ النحْوي مع معالجة مباشرة مقيدة. وفي عام 1981، كتبَ مؤلفو أداةٍ أولى لتحرير البِنى، "البرامج ليست نصوصًا؛ إنها تراكيبُ تراتبيةٌ لبِنى حسابية، وينبغي أن تُحرَّر وتنفَّذ وتُفلّى (debugged) في بيئةٍ تقبل بوجهة النظر هذه وتدعمها باستمرار." [35]

تساعِد أشكالُ الكتل المبرمجين المبتدئين على تحديد أيّ العبارات النحوية (التعبيرات أم الأوامر أم التصريحات) قانونية في سياقاتها. في لغة سكراتش، ترتبط الأوامر عموديًّا بنتوءات (nubs) وشقوق (notches)، في حين أن التعبيرات هي أشكال ملساء توضع في ثقوب ملساء. وتَحول القيودُ المطبقة على السحب والإفلات (drag-and-drop) دون الخلط بين هذين النوعين (الشكل 5). وقد أفاد الطلاب أن أشكال الألغاز مفيدة في تجميع البرامج. [38]

إنّ تمثيلَ الأنواع تمثيلًا مرئيًّا بالأشكال قابلٌ للتطبيق على أنظمةِ أنواعٍ أكثر ثراءً: فيوفِّر OpenBlocks أربعة عشرَ (14) شكلًا من الموصِّلات لتمثيل الأنواع المختلفة، [29] وأنشأ الباحثون لغاتِ كتلٍ تجريبيةً ذات أشكال تولِّد ديناميكيًّا لتمثيل أنظمةِ أنواعٍ تركيبية. [18، 36]

تُشجع الكتلُ التي يمكن التعامل معها مباشرة أيضًا على أداء الترقيع من الأسفل إلى الأعلى لِقطع البرامج بِطُرُقٍ غير مدعومة مباشرةً بالنص الخام. يقوم المبرمجون بلغة الكتل بتجريب الكتل وذلك بربطها لبناءِ جزر من قِطَعِ رمازٍ على سطحٍ



الشكل 5. توضِّح أشكالُ الكتل تركيبَ القواعد وتعزِّزها. تُركَّب أوامر سكراتش عموديًّا، وتوضع التعابير في فُتحات مناسبة. في هذا المثال، يُسحب تعبيرٌ بولياني (الشكل الماسي) ليوضع في فُتحةٍ موافِقةٍ لحالةِ اختبار الحلقة.

برمجيٍّ معزولٍ عن البرنامج الرئيسي. [22، 38] وفي بيئات الكتل الداعمة للرشاقة، [20] يمكن تنفيذ هذه القِطَع بالتأشير والنقر (pointing & clicking)، وهذا يوفِّر فائدةً من الفوائد الرئيسية التي تقدِّمها اللغات المعتمدة على النصوص المفسَّرة من دون مِعراض التحكم (اقرأ–قيِّم–اطبع) التكراريّ، المنفصلةِ عن المحرِّر. ينمو البرنامج تدريجيًّا بالإضافة وذلك بإسقاط القِطَع فيه عندما يكون سلوكها على النحو المرغوب فيه.

قابلية تعلُّم ما بعد الكتل. تساعِد الكتل على بناء الرماز، لكنها غير كافيةٍ بمفردها لجعل لغة البرمجة قابلة للتعلُّم. يواجه المبرمجون الجدد للغة تحدياتِ تعلُّمٍ إضافية:

- فعليهم أن يتدربوا على الجوانب العملية؛ مثل: تثبيت أدوات اللغة، وحفظ/تحميل البرامج، وما إلى ذلك؛
- وعليهم تعلّم مفردات اللغة واستنتاج المفاهيم التي تدل عليها كلماتها؛
- وهم بحاجة إلى معرفة مدلول وقت التنفيذ (runtime)؛ مثل: تدفق التحكم والتغيرات في الحالة بمرور الوقت؛
- وأخيرًا، هم بحاجة إلى تعلم أنماط الاستعمال الشائعة، والانتقال إلى ما بعد المفاهيم المعزولة.

يمكن تذليل (أو إزالة) كلّ عقبة من عقبات التعلم هذه بواسطة بيئة البرمجة، وتمثِّل كلُّ مسألة من هذه المسائل مجالَ بحثٍ وتطوير نشِط.

البرمجة الموصولة بالخط. لتبسيط عملية التثبيت، أصبحت أدوات البرمجة الموصولة بالخط (online). عندما تكون بيئة البرمجة في متصفح الوِب، يصبح المبرمج الجديد على بُعدِ نقراتٍ قليلة من إنشاءِ أولِ برنامج. يمكن أن توفِّر أداةٌ برمجيةٌ سحابية بيئةً برمجيةً كاملة ومتسقة ولا تحوي سوى عدد قليل من المشاكل المحتملة.

ومع أن بيئات البرمجة بالكتل للمبتدئين متاحة على الخط (online) منذ مدة طويلة، فإن بيئات البرمجة المعتمدة على النصوص قد أصبحت متاحة على الخط أيضًا. ويمكن للمبرمجين من جميع المستويات الاستفادة من العمل على الخط (online)، باستعمال أدواتٍ مثل Cloud 9، و CodeAnywhere، و CodeEnvy.

الكلمات والمفاهيم والتجريدات. يمكن أن يكون للأسماء المختارة للّبِنات اللغوية تأثير في قابلية التعلُّم. وقد وَجدَت الدراسات التجريبية التي قام بها Stefik and Seibert [34] أنّ الكلمات الرئيسية الشائعة مثل `for` أو المؤثِّرات مثل `!=` هي مصطلحاتٌ يصعب تعلُّمها، وليست بسهولةِ تعلُّم كلماتٍ مألوفةٍ مثل `repeat` أو `unequal`. ووجدا أن قابلية تعلمُّ التركيب النحْوي للغاتٍ مثل جافا وبيرل (Perl) ليست أشد منها في حالةِ لغة البرمجة النحْوية ذاتِ علاماتِ الترقيم المختارة عشوائيًّا للكلمات الرئيسية (keywords). وثَمة ما هو أشد أهميةً بخصوصِ قابلية التعلُّم وهو الأفكار التجريدية التي يختارها مصممو اللغة للسماح للمبرمجين ببناءِ برامج بسيطةٍ ذات سلوكٍ مقنِع. على سبيل المثال، عملَ مصممو أليسْ مع المبرمجين لتطوير تجريدات حدسية للتحكم في الرسوم المتحركة الثلاثية الأبعاد، وفي أثناء عملية التصميم الخاصة بهم، أزال فريق أليسْ مصطلحاتٍ مثل مصفوفات التحويل، ووضعوا بدلًا منها مفاهيم أكثر حدسية، مثل الحركات النسبية للأغراض، فكانت النتيجة أن هذه التجريداتُ الجديدة سهَّلت على المبرمجين تحديدَ الرسوم المتحركة الثلاثية الأبعاد. [7]

هذا ويعدّ تصميم اللغات والمكتبات الذي يركِّز على قابلية التعلُّم استنادًا إلى الأدلة التجريبية مجالًا رئيسيًّا للعمل في المستقبل.

فَهْمُ وقت التنفيذ. يمكن جعل الحالة الديناميكية للبرنامج أكثر قابليةً للفهم بجعل حالة البرنامج مرئية. على سبيل المثال، يسلِّط Code.org الضوءَ على الكتلة المنفردة الناشط تنفيذها، بحيث يمكن رؤية التقابل بين الرماز والإجراء المنفَّذ. وكذلك يوفر !Snap أدواتٍ لكل متغير بغية إظهار الحالة الحالية.

وقد يكون من الصعب فهم الإجراءات السابقة أو اللاحقة حتى مع وجود الحالة الواضحة جدًّا. والرشاقة (liveness) [20] هي أحد المناهج لمعالجة هذه المشكلة. إذ إن غاية النظام الرشيق (live) جعلُ الإجراءاتِ ملموسةً وذلك بتطبيقها على الفور على الحالة الحالية. ففي App Inventor، و Scratch، و Snap! مثلًا، يَظهر مفعولٌ فوريٌّ لكثير من التعديلات على البرنامج المعتمد على الكتل الذي يجري تنفيذه، ولا يحتاج الأمر إلى إعادة تشغيل البرنامج.

وثمة نهج آخر لجعل تطور الحالة قابلًا للفهم، وهو السماح للمبرمج بالتنقل مع الزمن، وذلك بمعاينة الخط الزمني للبرنامج، أو تحريكه نحو الأمام أو الخلف. وُصِف مفهومُ التقليةِ الكلّيةِ العِلم (omniscient debugging) أولَ مرة في أوائل السبعينيات باعتبارها قادرة على تتبعِ تاريخ التنفيذ رجوعًا في الزمن، وذلك لتعرّفِ موقعِ خطأ سبّبَ إخفاقًا لوحظ في وقت لاحق. [40] ويُذكر أن TRAKLA2 [26] و UUhistle [33] و Online Python Tutor [12] توفر هذه القدرة للُّغتين جافا وبايثن.

أمثلةٌ وإعادة الاستعمال. لقد تغيرت فعالية البرمجة بتوفر مستودعات كبيرة تحوي أمثلة لرمازٍ مشترَك، فصار المبرمجون على جميع مستوياتهم ينقلون ما يجدونه من أمثلة ويكيفونها لتكون فعاليةً برمجيّة جوهرية. [6، 9] واستجابةً لذلك، بدأت بيئاتُ البرمجة الاحترافية وبيئاتُ المستعمل النهائي توفِّر الدعمَ بالأمثلة. [5، 31] ويرغب المبتدئون أيضًا، في أن يتعلموا بالأمثلة، لكنهم قد يجدون صعوبة في تحقيق ذلك. [30]

تَلجأ اللغاتُ المعتمدة على الكتل – مثل سكراتش ولوكينغ غلاس (Looking Glass) – إلى المشاركة الموصولة بالخط (online) وإعادة دمج البرامج لتوفير النفاذ إلى الأمثلة. لكنْ ثَمة موازنة بين بساطةِ إعادة الاستعمال ومتانةِ الرماز الذي يُعاد استعماله. على سبيل المثال، تُبسِّط سكراتش التشاركَ في أمثلة الرماز للمبتدئين بتوفير "حقيبة ظَهر" لجمع مقتطفاتِ البرامج والأصولِ التي يمكن المشاركة فيها وإدراجها إلى مشروعٍ جديد. ومع ذلك، فإن حقيبة الظَّهر لا تضْمن بالضرورة تنفيذ الرماز تنفيذًا صحيحًا في مشروعٍ جديد.

وفي المقابل، تَلجأ لوكينغ غلاس إلى استعمال إجرائيةٍ أكثر تعقيدًا فيما يتعلق بإعادة الاستعمال، بحيث يختارُ المبرمجون بدايةَ السلوك الذي يرغبون في استعماله ونهايَته. فإذا ما قرن هذا بمعلوماتِ سجل التنفيذ، أمكن ضمان تنفيذِ الرماز المختار ضمن سياقِ برنامجٍ جديدٍ. هذا وتَسمح ميْزةُ شغِّل واستكشف (Play & Explore) للمبرمجين بربط خرج البرنامج بسطرِ (أو أسطرِ) الرماز التي سبَّبت ذاك الخرج، وهذا يساعِد المبرمجين على فهْمِ الرماز الذي أُعيدَ استعماله والبدءِ في تعديله.

توسيع رماز الكتل

لماذا لا يبرمِج المحترفون باستعمال واجهات الكتل؟ إن أحد الأسباب هو أن للتعامل المباشر عيوبًا في الكفاءة عند إجراء تعديلات طفيفة. فعند إنشاءِ تعبيرٍ مثل $(a / 2 + b / 2)$ بالكتل، سيكون على المبرمج أن يبحث عن الكتل ويسحبها لكلٍّ من المؤثِّرات الحسابية الثلاثة، ثم يملأ الفراغات بالمتغيرات والأرقام. وبالمثل، عند إعادةِ ترتيبِ تعبيرٍ من $(a / 2 + b / 2)$ إلى $(a + b) / 2$، يجب سحب شجرة التعبير جانبًا وإعادة تجميعها ثانية، وهذا يتطلب إيماءاتٍ أكثر وتدبُّرًا أكبر من إجراء التعديلات في النص. وقد لاحظ باحثو HCI[4] أن لغات البرمجة المرئية يمكن أن يكون لها *لزوجة* (viscosity) أعلى منها في حالة الرماز النصي، وذلك لأنها تجعل التغييرات الصغيرة أكثر صعوبة. [11]

وإضافةً إلى اللزوجة، قد يكون لبيئاتِ الكتل عدة عيوب في قابلية الاستعمال مقارنةً بلغات البرمجة النصية؛ منها:

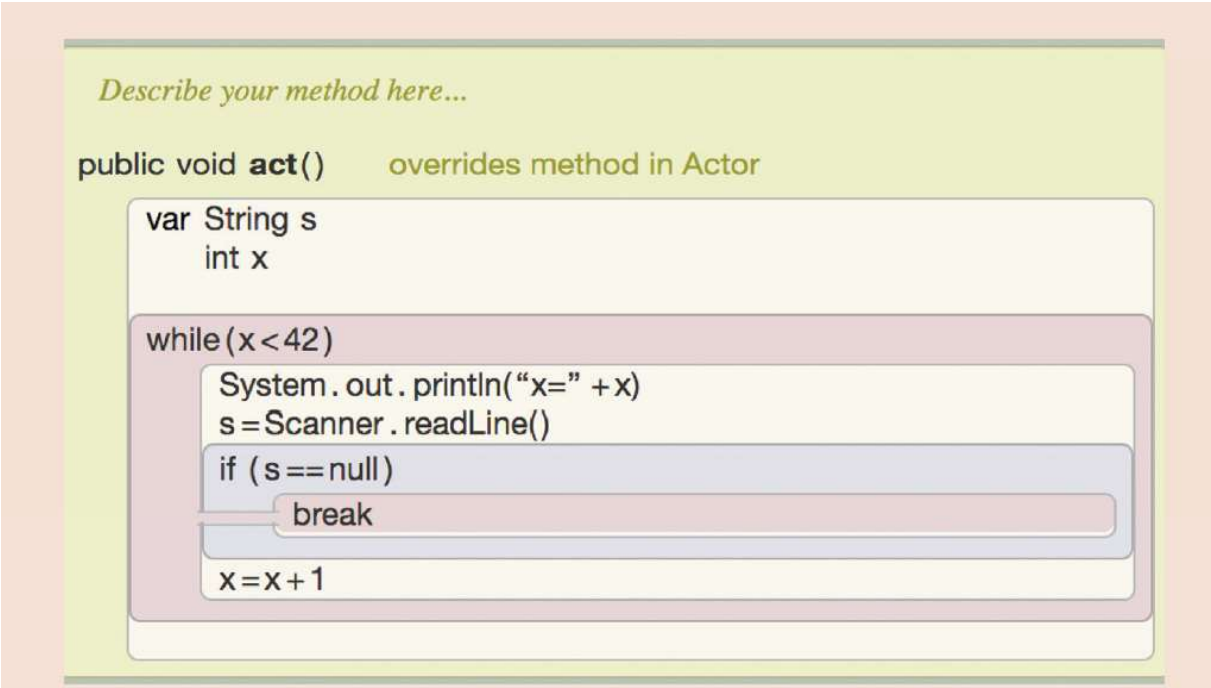
- الكثافة المنخفضة: تَشغل الكتل مساحةً على الشاشة أكبر مما يشغله رمازِ النصّ المكافئ.
- البحث والتنقل: قد يصعب العثور على الجزء الذي نبحث عنه في برنامجِ الكتل، ثم الانتقال إليه في مساحةِ عملٍ ثنائيةِ البُعد قد لا تكون مرئيةً بتمامها.
- التحكم في المصدر: من الصعب استعمال أنظمة التعاون والتحكم في الإصدارات من دون تمثيلٍ نصّي للرماز.

يتضمن الجيلُ الأحدث من أدوات برمجة الكتل مِيْزاتٍ صُمِّمت لإيجاد تناغم بين مزايا استعمال النص ومزايا استعمال الكتل. ثَمة منهجان: الإدخال النصي، والتبديل الثنائيّ الاتجاه للنمط.

الإدخال النصي للكتل. صُمِّمت بيئاتُ كتل جديدة – مثل المحررِ Stride المعتمد على الإطارات، من القدم الخضراء (Green-foot) [16] ولغةِ GP [24] – للمبرمجين لإنشاء برامج ضخمة، وكان التحرير الفعّال إحدى غايات التصميم الهامة.

يحسِّن كلّ من Stride و GP الكفاءةَ بتوفيرِ اختصاراتِ التحرير المعتمد على النص ضمن واجهةٍ كتلية التوجُّه (blocks-oriented interface). ولتمكين المبرمجين من التغلب على خطوة العثور على كتلة ما في لوحة الكتل، يتيح هذان النظامان للمبرمجين إدراجَ الكتل بآليةِ الإكمال التلقائي المضمَّنة (in-line autocomplete). ويمكن دومًا اختيار الكتل من لوحة الكتل، إلا أن بمقدور المبرمج اللبيب إدراج الكتل عن طريق الكتابة. يقدِّم المحررُ Stride أيضًا منهجًا هجينًا لتحرير الرماز، يميّز بين البِنية ذات المستوى المنخفض والبِنية ذات المستوى العالي (الشكل 6). ففي حالة الرماز على مستوى التعبير (expression-level code)، يُخفي البنيةَ النحْويةَ ويَسمح بالتعديل التقليدي للنص، معطيًا إظهارًا عالي الكثافة ولزوجةً

[4] Human Computer Interface: الواجهة بين الحاسوب والإنسان.



الشكل 6. يَجمع محررُ Stride من Greenfoot بين التحرير بالنمطِ النصيّ عندما تكون التفاصيل على مستوى التعبير وبين التحرير بنمط سحب وإفلات الكتل عندما تكون بِنية البرنامج ذات مستوى عال.

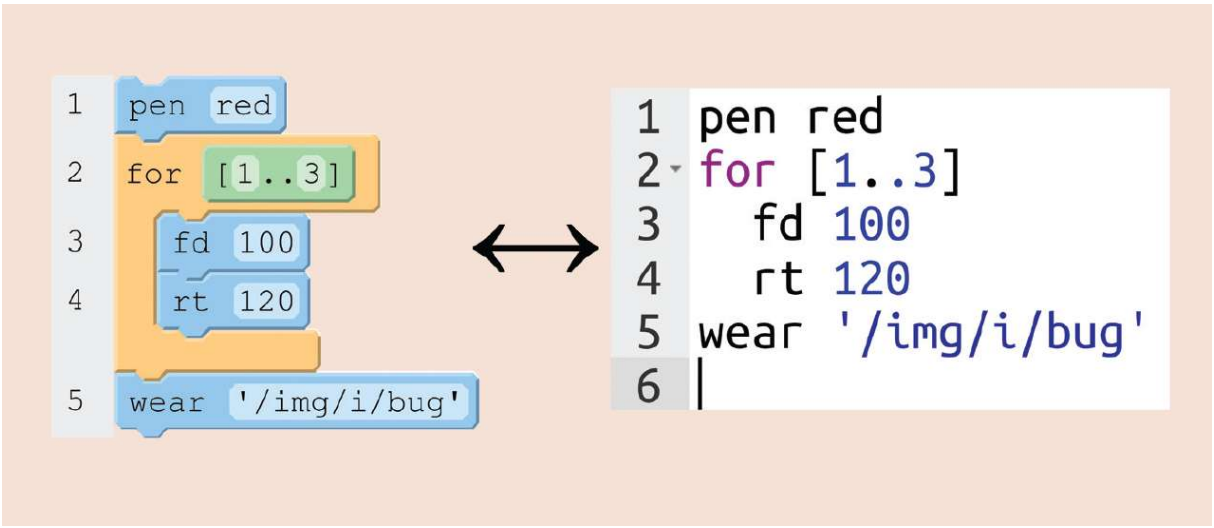
منخفضة. ويُلجأ إلى بِنيةٍ شجريةٍ مرئيةٍ ومعالجةٍ بالسَّحْب والإفلات في حالةِ رمازٍ من مستوٍ أعلى، مثل رمازِ التدفق في التحكم والتصريح عن الصفوف.

التبديل الثنائيّ الاتجاه للنمط. توفِّر بعضُ بيئات الكتل تحويلًا ثنائيَّ الاتجاه بين لغة نصية تقليدية وتمثيلِ الكتلِ لهذه اللغة. ومن هذه البيئات: Pencil Code (CoffeeScript) [2] (الشكل 7)، و App Lab من موقع Code.org [5] (جافا سكريبت)، و BlockEditor (جافا) [21]، و Tiled Grace (Grace). [13]. أما أليسْ وبلوكلي فيقدّمان رؤيةً غير قابلة للتعديل للرماز النصي.

الفرضية التي تحضُّ على تصميمَ أدواتِ النمطِ المزدوج هي أن المبرمجين يمكن أن يستفيدوا من قابليةِ تعلّمِ الكتل في نمطٍ ما، على حين يتعلمون التركيبَ النحْوي ويستفيدون من كفاءة النص في النمط الآخر. وتتطلب هذه الغاية ربطَ الرؤيتَين إحداهما بالأخرى. فلكي يكون النص سليمًا للمبرمجين الذين قد يرغبون في العودة إلى الكتل، يجب أن تكون المبادلة بين النمطين ممكنة.

في المحررات ذات النمط المزدوج (dual-mode editors)، يكون الرمازُ النصيُّ هو التمثيلَ الأساسي للبرنامج، وتكون الكتل مرأى واجهةِ المستعمل المتوقعة المستنبطة من التحليل النحْوي (parsing). يُتيح هذا النهجُ للمحرِّر تمثيلَ معلوماتِ النص تمثيلًا كاملًا، كالتباعد مثلًا، لكنه يعني أيضًا أن على المحرر أن يجيز الأخطاء النحْوية التي لم تكن ممكنة

[5] https://code.org/educate/applab



الشكل 7. يوفِّر Pencil Code تبديلًا ثنائيَّ الاتجاه بين الكتل والنص. يتيح تبديلُ النمط للمبرمجين تعلُّمَ الكتل وتحريرًا سريعًا للنص.

في الكتل. وتقوم طرائق استعادة الأخطاء التجريبية على تحويل الأخطاءِ النحْوية البسيطة في النص إلى أخطاء خاصة بالكتل، أما الأخطاء المعقدة فيمكنها أن تمنع تبديل الأنماط.

مقارنة النهجين. ثَمة مقايضة بين النهجين لتوحيد الكتل والنص. ففي حين توفِّر المحرِّرات ذات النمط المزدوج دعمًا مباشرًا لتعلُّم التركيب النحْوي التقليدي للنص، مثل جافا سكريبت أو جافا، فإنها أيضًا تَفرض أعباءً معرفية للتعامل مع الأخطاء النحْوية التي لا يمكن أن توجد إلا في نمط النص. وتشير بحوثُ الإظهار المرئي الخاصة بالرؤى المتعددة المنسَّقة إلى فوائد توفيرِ أكثر من رؤيةٍ واحدة لمقابلة الأعباء المعرفية التي يفرضها التبديل بين الرؤى. [37]

لمحررات البنية المنفردة النمط مِيْزةُ خُلوِّ النموذجِ المفاهيميّ من أخطاء التركيب النحْوي، لأن الكائن الأساسي الخاضع للتحرير هو شجرة التركيب النحْوي المجرد. ومع ذلك، للحفاظ على التناسق، يجب منع العديد من أنواع التعديلات النصية، ويمكن أن تتطلب تعديلاتٌ أخرى أوامر خاصة لتحرير الأشجار. وهذه القيود تزيد من لزوجة التحرير، ويمكن أن تسبِّب عوائقَ معرفيةً إضافية. هذا وتُعَدّ الواجهات التي تَسدّ الفجوة بين الكتل والنص بفعالية مجالًا رحبًا للبحث.

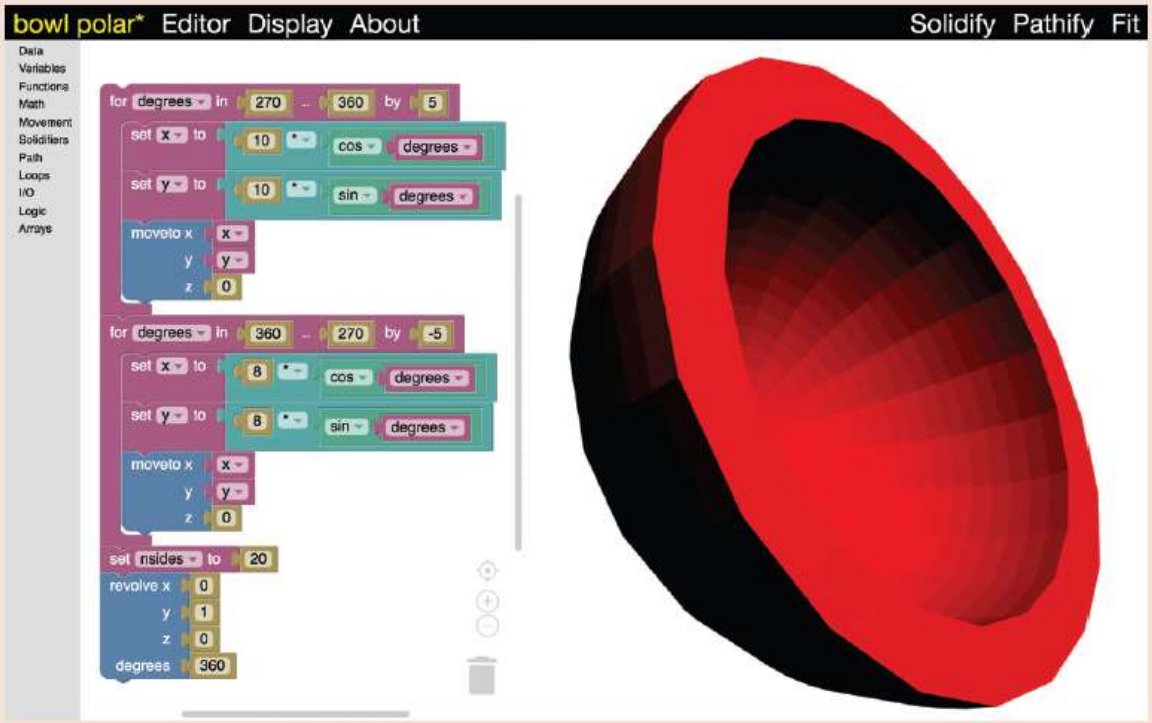
تطبيق الكتل: مثالان

دعونا نلقِ نظرةً على لغاتِ كتلٍ تجريبية من مجالين محدَّدين غير مألوفين لمعظم المبرمجين.

برمجة الطابعات الثلاثية الأبعاد. تقليديًّا، تُرسم نماذج الطباعة الثلاثية الأبعاد (3D) تفاعليًّا بالمعالجة المباشرة باستعمال برمجية CAD. وتُعَدّ كتابةُ رمازٍ مخصَّصٍ بغية إنشاءِ نموذجٍ ما منهجًا بديلًا فعالًا، بيد أنّ كتابة الرماز لصناعةِ أشكالٍ ثلاثية الأبعاد كان تقليديًّا عالمَ عدد قليل من المبرمجين الخبراء. أما الآن، وبسبب انخفاضِ أسعار الطابعات الثلاثية الأبعاد وتوفرِها المتزايد، فأصبح بمقدور غير المتخصصين كتابةُ رماز النمذجة الثلاثية الأبعاد المناسب لطلبهم.

جَلبتْ اثنتان من اللغات المنجزَة حديثًا البرمجةَ المتعلقة بالطباعة الثلاثية الأبعاد إلى المبرمجين المبتدئين: BeetleBlocks (خنافس الكتل) [17] و Madeup. [14] ومع أن لهذين النظامين لغتين مختلفتين، فإنّ لهما قواسم مشتركة: فلكليهما واجهةٌ معتمِدة على الوِب تَسمح بالتصيير الثلاثي الأبعاد للشكل الذي يجري إنشاؤه، وكلاهما يوفِّر لغةَ كتلٍ بغية تبسيط تعلُّمها. تتْبع اللغة BeetleBlocks مبادئَ بيانياتِ السلاحف: تُحرِّك الخنفسةُ "قلمًا" يمكِن تشغيله وإيقاف تشغيله، ويمكِن إنشاءُ الأشكال الثلاثية الأبعاد من جرّات القلم المتكرّرة. وتَتخذ اللغةُ MadeUp (الشكل 8) منهجًا أكثر تجريدًا، إذْ تتيح للمبرمجين تتبُّعَ المساراتِ والسطوحَ الوسيطية[6] على حدّ سواء. وتتيح وظائفُ خاصةٌ تدويرَ المسارات أو بثّها لإنشاء المجسمات.

تقدم اللغتان مستوياتٍ مختلفة من القدرة والتجريد. لكنْ، أيُّ لغةٍ منهما أصلح للاستعمال؟ هذا المجال هو مثال ممتاز لِمِيْزة قابليةِ تعلُّم الكتل. فلكلتا اللغتين منحني تعلُّم ضحلٌ جدًّا، ومن السهل تجريبهما.



الشكل 8. برمجة الكتل في بيئة MadeUp. تُعَدّ الطباعة الثلاثية الأبعاد أحدَ مجالات الابتكار السريعة، وتمكِّن الكتلُ من استعمالِ لغاتِ النمذجة الثلاثية الأبعاد الجديدة دون وجودِ منحني تعلّمٍ شديد الانحدار.

[6] parametric surfaces: هي وظائف لها دخل ثنائي البعد وخرج ثلاثي الأبعاد، من قبيلِ رسمِ سطحٍ في فضاءٍ ثلاثي الأبعاد. (المترجم)

استعلام الوِب الدلالي. تُعَدّ الحاجةُ المتكررة إلى استعلامِ مجموعاتِ المعطيات الضخمة مجالًا آخر حيثما تكون قابلية التعلُّم جوهرية. تأمّل مسألة استعلامِ معطياتِ إطارِ عملِ وصفِ الموارد (Resource Description Framework, RDF) من الوِب الدلالي. إن لغة SPARQL هي اللغة المعيارية لاستعلامِ معطياتِ RDF؛ وتتضمن العديدَ من اللبِناتِ للتعامل مع ثلاثياتِ RDF التي تُميزها من لغات الاستفسار الأخرى مثل SQL. ومع ذلك، يواجه مستعملو لغة SPARQL المحتملين عقبتين: الأولى، أن على المبرمجين أن يتعلموا مفرداتِ SPARQL وتركيبَها النحْوي مع مؤثِّراتها ولبِناتها المتخصصة. والثانية، أن استعلام معطياتِ RDF لا يتطلب معرفةَ اللغة فحسب، بل يتطلب أيضًا معرفةَ الأمثال (instances) ومعطياتِ المُخْتطات (schemas).

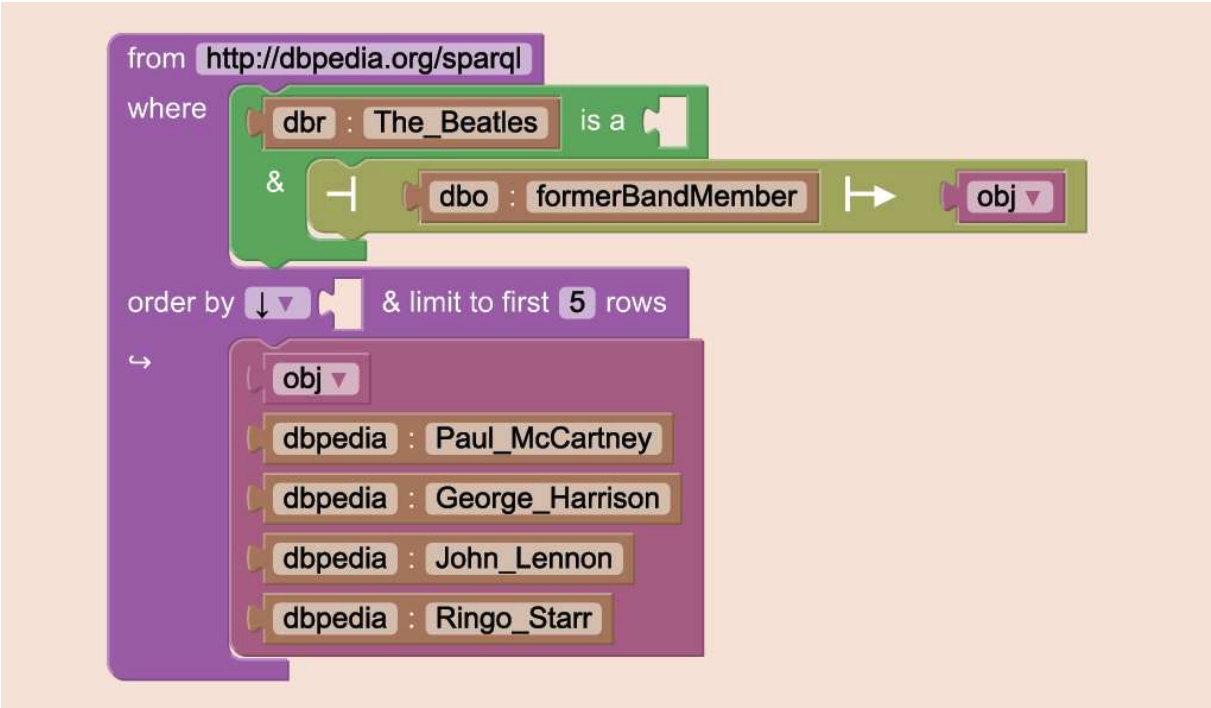
ولمعالجة كلتا المشكلتين، أنشأ باولو بوتوني وميغيل سيرياني (Paolo Bottoni & Miguel Ceriani) لغةَ كتلٍ سمَّوْها *SPARQL Playground* (ملعب سباركل). [4] وبدلًا من أن تساعد لغةُ الكتل الخاصة بهما المبرمجين على تعلُّم البرامج التتابعية، فإنها تساعدهم على اختيار المعطيات وتصفيتها وضمّها (select, filter, join) باستعمال بدئيات SPARQL.

وثمة سبب آخر يجعل لغةَ ملعب سباركل مثيرة للاهتمام؛ وهو أن جميع نتائج الاستعلام المحصَّلة في "الملعب" تعاد أيضًا في هيئةِ كتلٍ قابلة للسحب (الشكل 9). تتيح هذه المِيْزة للمبرمجين حفظَ معطياتِ الأمثال (instance data) في مساحةِ عملِ البرمجة كي تُدمَج في استعلامات جديدة. ومن السهل، مع ملعب سباركل، البدءُ باستعلاماتٍ عامة لاستكشافِ أنواع المعطيات المتاحة، ثم استعمالُ هذه المكتشَفات لإجراء التحسينات المرغوب فيها.

صنع لغاتٍ جديدةٍ للكتل

أصبح بوسع المبرمج إنشاء بيئةِ كتلٍ مخصَّصةِ المجال خاصة به باستعمالِ طقمِ أدواتٍ للّغةِ المعتمدة على الكتل. وينبغي أن يكون مؤلفو لغةِ الكتل على درايةٍ بثلاثة أطقمِ أدوات على الأقل: بلوكلي[7]، ودروبلت، [3] و OpenBlocks. [29] إنّ لغةَ الكتل الوصفية (metalanguage) الأصلية هي OpenBlocks من MIT، وقد اخترعها ريكاروز روك (Ricarose Roque) في عام 2007 لتكون أساسًا للغة StarLogo TNG، وتمتاز بقدرٍ كبيرٍ من المواءمة الهندسية حسب الطلب. جرى أيضًا استعمال OpenBlocks في App Inventor Classic و BlockEditor. ويتمثل أحد عيوب OpenBlocks في أنه يتطلب من المبرمجين تنزيل حزمة Java JDK وتثبيتها. جرت معالجة مسألة التثبيت بواسطة بلوكلي، وهي طقم أدواتٍ للغةِ كتل معتمدة على HTML أعدَّه نيل فرازر (Neil Fraser) من شركة غوغل. ويُعَدّ بلوكلي حاليًا أشيعَ طقمِ أدواتٍ لغاتِ الكتل: فهو أداة في App Inventor، وملعب SPARQL، و MadeUp، إضافةً إلى ألغازِ *ساعة في الرماز* (Hour of Code) لموقع Code.org. هذا وإن الإصدارات المستقبلية من سكراتش ستَستعمل بلوكلي أيضًا. يُعَدُّ دروبلت أحدث طقمِ أدواتٍ لإنشاء لغات الكتل، أنشأه أنتوني باو (Anthony Bau) لاستعماله في بيئةِ Pencil Code، وهو مستعمَل أيضًا في App Lab في موقع Code.org، وهو أحدث وأقل نضجًا من OpenBlocks وبلوكلي، لكنه يأخذ منهجًا فريدًا يتيح تحويلًا سلسًا ثنائيَّ الاتجاه بين كل الكتل والرماز النصي.

[7] https:developers.google.com/blockly



الشكل 9. ملعب SPARQL هو أداةٌ لتنفيذِ استعلاماتٍ معتمدة على الكتل، توفِّر كتلًا لإنشاءِ استعلاماتِ معطياتِ RDF، وتعطي نتائجَ الاستعلام (أسفل) على شكلِ كتل أيضًا، بحيث يمكن سحبُها لإنشاءِ استعلاماتٍ أخرى.

الخلاصة

عندما تُقدَّم لغةُ البرمجة بصفتها واجهةً للمبرمج ترحب بالمبرمجين المبتدئين، لا بصفتها أداةً تقنية لمطوِّرين ذوي خبرة فحسب، فإننا نتوصل إلى صورةٍ جديدة لما يجب أن توفره بيئة البرمجة:

- ينبغي أن تُستمد المفردات من التمييز، لا من التذكر؛
- ينبغي أن يقلَّل الحِمل المعرفي بتقطيع الرماز؛
- ينبغي أن تُجعل قواعد النحو وأنواعها مرئية؛
- ينبغي أن تكون قِطَع البرامج قابلة للتعامل المباشر؛
- ينبغي أن يكون التحرير المنخفض اللزوجة ممكنًا أيضًا؛
- ينبغي أن تكون بيئات الترميز متاحة دون الحاجة إلى تثبيت الأدوات؛
- ينبغي وصفُ المفاهيم البسيطة بكلماتٍ واضحةٍ وتجريداتٍ عالية المستوى؛
- ينبغي أن تكون الحالة والسلوك في وقت التنفيذ مرئية؛
- ينبغي أن يكون العثورُ على الأمثلة وتطبيقُها سهلًا.

وبالجملة، لكي تكون أداة البرمجة قابلة للاستعمال من المبرمجين الجدد أو العرضيين، يجب أن يركز تصميمها على قابلية التعلُّم. وقد أَثبتت الكتل أنها فعالة في حل العديد من هذه المسائل. ومع أنه لا يزال تعلُّم البرمجة على نطاقٍ واسع ليس كما ينبغي، فإن التقدم الذي أحرزته واجهات لغاتِ الكتل يمكن أن يكون مَصدر إلهامٍ لنا جميعًا لنرى أنه يمكن جعل البرمجة أكثر قابلية للتعلُّم. إن فن البرمجة هو التفاعل الأصلي بين الإنسان والحاسوب، ويظل تحديًا على صعيدِ قابلية الاستعمال لا حل له بعد. ولا يزال بوسعنا عمل المزيد لجعل البرامج متاحة للجميع.

أدوات تعليمية معتمدة على الكتل

على مواقع الوِب:

- سكراتش (https://scratch.mit.edu): رسوم متحركة وإنشاء اللُّعب في المتصفح، مع دعم للتوسّعات
- Code.org (http://code.org): مجموعة متنوعة من الأدوات، ومنها تمارين برمجة الألغاز مع مقاطع فيديو تعليمية
- Snap! (http://snap.berkeley.edu): لغة محسنة مستوحاة من سكراتش تتضمن وظائف من الدرجة الأولى
- App Inventor (http://appinventor.mit.edu): إنشاء تطبيقات أندرويد باستعمال بيئة تطوير متكاملة (IDE) للكتل في المتصفح
- Pencil Code (https://pencilcode.net): ينشئ تطبيقات وب باستعمال لغة Coffee-Script، للتحويل بين النص والكتل
- StarLogo Nova (http://www.slnova.org): محاكاة متعددة الوكلاء ولُعَب في عالمٍ مصيَّرٍ ثلاثي الأبعاد
- لُعَب بلوكلي (https://blocklygames.appspot.com): مجموعة من الألغاز لحلها باستعمال كتل البرمجة
- GameBlox (https://gameblox.org): إنشاء لُعَب تتضمن وكلاء قابلة للانتساخ، وعلوم فيزيائية، وغيرها

قابلة للتنزيل:

- AgentSheets/AgentCubes (http://www.agentsheets.com): بيئات كتل رائدة لإنشاء لُعَبٍ تعتمد على القواعد (rules) والمحاكاة
- أليس (http://www.alice.org): بيئة كتل رائدة لإنشاء عوالم افتراضية ثلاثية الأبعاد؛ توفّر دعمَ التصدير إلى جافا
- Looking Glass (https://lookingglass.wustl.edu): بيئة لإنشاء قصةٍ متحركةٍ ثلاثية الأبعاد؛ تَدعم التعلُّم المستقل
- كودو (http://www.kodugamelab.com): برمجة معتمدة على القواعد (rules) لإنشاءِ لُعَبٍ لأجهزةِ xBox والحاسوب الشخصي

على الجوّال:

- سكراتش جي آر (http://www.scratchjr.org): برمجة المَشاهد المتحركة الموجَّهة للأطفال لما قبل مرحلة التعليم
- Pocket Code (http://www.catrobat.org): برمجة الكتل للتجهيزات الجوالة الصغيرة الحجم
- Tynker (https://www.tynker.com): منصة تجارية قشيبة لإنشاء اللُّعَب والرسوم المتحركة
- Hopscotch (https://www.gethopscotch.com): إنشاء ألعاب ورسوم متحركة على أجهزة iPhone و iPad

المراجع

[1] Armoni, M., Meerbaum-Salant, O. and Ben-Ari, M. From Scratch to 'real' programming. Trans. Computing Education 14, 4 (Feb. 2015).

[2] Bau, D., Bau, D.A., Dawson, M., and Pickens, C.S. Pencil Code: Block code for a text world. In 14th International Conference on Interaction Design and Children, (2015), 445–448.

[3] Bau, D.A. Droplet, a blocks-based editor for text code. J. Computing Sciences in Colleges 30, 6 (June 2015), 138–144.

[4] Bottoni, P. and Ceriani, M. Using blocks to get more blocks: Exploring linked data through integration of queries and result sets in block programming. In IEEE Blocks and Beyond Workshop, Oct. 2015, 99–102.

[5] Brandt, J., Dontcheva, M., Weskamp, M. and Klemmer, S.R. Example-centric programming: Integrating Web search into the development environment. In Proceedings of ACM SIGCHI Conference on Human Factors in Computing Systems, 2010, 513–522.

[6] Brandt, J., Guo, P.J., Lewenstein, J., Dontcheva, M., and Klemmer, S.R. Two studies of opportunistic programming: Interleaving Web foraging, learning, and writing code. In Proceedings of ACM SIGCHI Conference on Human Factors in Computing Systems, 2009, 1589–1598.

[7] Conway, M., Audia, S., Burnette, T., Cosgrove, D. and Christiansen, K. Alice: Lessons learned from building a 3D system for novices. In Proceedings of ACM SIGCHI Conference on Human Factors in Computing Systems, 2000, 486–493.

[8] Dann, W., Cosgrove, D., Slater, D., Culyba, D. and Cooper, S. Mediated transfer: Alice 3 to Java. In 43rd ACM Technical Symposium on Computer Science Education, 2012, 141–146.

[9] Dorn, B. and Guzdial, M. Graphic designers who program as informal computer science learners. In Proceedings of 2nd International Workshop on Computing Education Research, 2006, 127–134.

[10] Garlick, R. and Cankaya, E.C. Using Alice in CS1: A quantitative experiment. In 15th Annual Conference on Innovation and Technology in Computer Science Education, 2010, 165–168.

[11] Green, T.R.G . Cognitive dimensions of notations. People and Computers V. A. Sutcliffe and L. Macaulay, Eds. Cambridge University Press, Cambridge, UK, 1989, 443–460.

[12] Guo, P.J. Online Python tutor: Embeddable Webbased program visualization for CS education. In Proceedings of the 44th ACM Technical Symposium on Computer Science Education, 2013, 579–584.

[13] Homer, M. and Noble, J. Combining tiled and textual views of code. In Proceedings of the 2014 IEEE Working Conf. Software Visualization, Sept. 2014, 1–10.

[14] Johnson, C. and Bui, P. Blocks in, blocks out: A language for 3D models. IEEE Blocks and Beyond Workshop, Oct. 2015, 77–82.

[15] Ko, A.J., Myers, B.A. and Aung, H.H. Six learning barriers in end-user programming systems. In Proceedings of the IEEE Symp. Visual Languages and Human Centric Computing, 2004, 199–206.

[16] Kolling, M., Brown, N.C.C., and Altadmri, A. Framebased editing: Easing the transition from blocks to text-based programming. In Proceedings of the 10th Workshop in Primary and Secondary Computing Education, Nov. 2015.

[17] Koschitz, D. and Rosenbaum, E. Exploring algorithmic geometry with 'Beetle Blocks:' A graphical programming language for generating 3D forms. In Proceedings of the 15th International Conference on Geometry and Graphics, Aug. 2012, 380–389.

[18] Lerner, S., Foster, S.R., and Griswold W.G. Polymorphic blocks: Formalism-inspired UI for structured connectors. In Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems, 2015, 3063–3072.

[19] Malan, D.J. and Leitner, H.H. Scratch for budding computer scientists. ACM SIGCSE Bulletin 39, 1 (Mar. 2007), 223–227.

[20] Maloney, J.H. and Smith, R.B. Directness and liveness in the morphic user interface construction environment. In Proceedings of the 8th Annual ACM Symposium on User Interface and Software Technology, 1995, 21–28.
[21] Matsuzawa, Y., Ohata, T., Sugiura, M. and Sakai, S. Language migration in non-CS introductory programming through mutual language translation environment. In Proceedings of the 46th ACM Technical Symposium on Computer Science Education, 2015, 185–190.
[22] Meerbaum-Salant, O., Armoni, M. and Ben-Ari, M. Habits of programming in Scratch. In Proceedings of the 16th Annual Joint Conference on Innovation and Technology in Computer Science Education, 2011, 168–172.
[23] Miller, G.A. The magical number seven, plus or minus two: some limits on our capacity for processing information. Psychological Review 63, 2 (1956), 81.
[24] Monig, J., Ohshima, Y. and Maloney, J. Blocks at your fingertips: Blurring the line between blocks and text in GP. IEEE Blocks and Beyond Workshop, 2015, 51–53.
[25] Moskal, B., Lurie, D. and Cooper, S. Evaluating the effectiveness of a new instructional approach. ACM SIGCSE Bulletin 36, 1 (2004), 75–79.
[26] Nikander, J., Korhonen, A., Seppala, O., Karavirta, V., Silvasti, P. and Malmi, L. Visual algorithm simulation exercise system with automatic assessment: Trakla2. Info. Education—An International J. 32 (2004), 267–288.
[27] Powers, K., Ecott, S. and Hirshfield, L.M. Through the looking glass: Teaching CS0 with Alice. ACM SIGCSE Bulletin 39, 1 (Mar. 2007), 213–217.
[28] Resnick, M., Maloney, J., Monroy-Hernلndez, A., Rusk, N., Eastmond, E., Brennan, K., Millner, A., Rosenbaum, E., Silver, J., Silverman, B., and Kafai, Y. Scratch: Programming for all. Commun. ACM 52, 11 (Nov. 2009), 60–67.
[29] Roque, R. Openblocks: An extendable framework for graphical block programming systems. Master's thesis, MIT, May 2007.
[30] Rosson, M.B., Ballin, J. and Nash, H. Everyday programming: Challenges and opportunities for informal Web development. In Proceedings of the 2004 IEEE Symposium on Visual Languages and Human Centric Computing, Sept. 2004, 123–130.
[31] Sawadsky, N. and Murphy, G.C. Fishtail: From task context to source code examples. In Proceedings of the 1st Workshop on Developing Tools As Plug-ins, 2011, 48–51.
[32] Shapiro, R.B. and Ahrens, M. Beyond blocks: Syntax and semantics. Communications of the ACM 59, 5 (May 2016), 39–41.
[33] Sorva, J. and Sirkia, T. Context-sensitive guidance in the UUhistle program visualization system. In Proceedings of the 6th Program Visualization Workshop, 2011, 77–85.
[34] Stefik, A. and Siebert, S. An empirical investigation into programming language syntax. Trans. Computing Education 13, 4 (Nov. 2013), 19:1–19:40.
[35] Teitelbaum, T. and Reps, T. The Cornell program synthesizer: A syntax-directed programming environment. Commun. ACM 24, 9 (Sept. 1981), 563–573.
[36] Vasek, M. Representing expressive types in blocks programming languages. Undergraduate thesis, Wellesley College, May 2012.
[37] Wang Baldonado M.Q., Woodruff, A., and Kuchinsky, A. Guidelines for using multiple views in information visualization. In Proceedings of the Working Conference on Advanced Visual Interfaces, 2000, 110–119.
[38] Weintrop, D. and Wilensky, U. To block or not to block, that is the question: students' perceptions of blocks-based programming. In Proceedings of the 14th International Conference on Interaction Design and Children, 2015, 199–208.
[39] Wolz, U., Leitner, H.H., Malan, D.J., and Maloney J. Starting with Scratch in CS1. ACM SIGCSE Bulletin 41, 1 (Mar. 2009), 2–3.
[40] Zelkowitz, M. Reversible execution. Commun. ACM 16, 9 (Sept. 1973), 566.

المؤلفون

ديفيد باو (David Bau) (davidbau@csail.mit.edu) طالب دكتوراه في مختبر علوم الحاسوب والذكاء الصنعي بمعهد ماساتشوستس للتقانة، كامبريدج، ماساتشوستس.

جيف جري (Jeff Gray) (gray@cs.ua.edu) أستاذ في قسم علوم الحاسوب في جامعة ألاباما، توسكالوسا.

كيتلن كيليهر (Caitlin Kelleher) (ckelleher@cse.wustl.edu) أستاذ مساعد في قسم علوم الحاسوب في جامعة واشنطن، سانت لويس، MO.

جوش شيلدون (Josh Sheldon) (jsheldon@csail.mit.edu) مدير البرامج و App inventor منMIT في مختبر علوم الحاسوب والذكاء الاصطناعي بمعهد ماساتشوستس للتكنولوجيا، كامبريدج، ماساتشوستس.

فرانكلين توربك (Franklyn Turbak) (fturbak@cs.wellesley.edu) أستاذ مشارك في قسم علوم الحاسوب في كلية ويلسلي (Wellesley)، ويلسلي، ماساتشوستس.

البروفسكيتات تُعزِّز إمكانات الخلايا الشمسية

PEROVSKITES BOOST SOLAR-CELL POTENTIAL*

Don Monroe
ترجمة: د. أحمد حصري
مراجعة: د. خالد مصري

قد تتيح موادُ جديدةٌ إنتاجَ خلايا شمسية أقل تكلفةً وأحسن مردودًا للتطبيقات التقليدية والمستحدثة.

حدث تقدمٌ سريعٌ في التقانة الفوتوفولطائية[1] خلال السنوات الخمس المنصرمة، زاد من تسارعه استعمالُ موادَّ تُسمى بروفسكيتات[2] (perovskites). تتطلب هذه المواد مكوّناتٍ شائعةٍ فقط، وطرائق تصنيعٍ سهلةٍ نسبيًا، وهذا ما يُتيح إمكانَ تصنيع خلايا شمسية ذات أغشيةٍ رقيقة رخيصة على سطوحٍ متنوعة، أو ضمّها مع السليكون في ألواحٍ كبيرة. لقد أمكن مخبريًا بالفعل تصنيع خلايا صغيرة المساحة من هذه المواد بلغ مردود تحويلها للطاقة الشمسية 22%، منافسةً بذلك مردود الخلايا الشمسية السليكونية التقليدية.

يقول دافيد كاهن (David Cahen)، وهو كيميائي وعالم مواد من معهد وايزمان في فلسطين المحتلة، "إنني أعمل في هذا المجال منذ سبعينيات القرن الماضي، وقد آن الأوان لإحداث تطوراتٍ غير مسبوقة في الخلايا الشمسية والتقانة الفوتوفولطائية، تشمل استعمال موادَّ جديدةٍ وتحسين المردود وتخفيض التكلفة." ويضيف " لم يسبق أن حدث شيءٌ كهذا." لقد تفوقت البروفسكيتات في كل هذه الأمور".

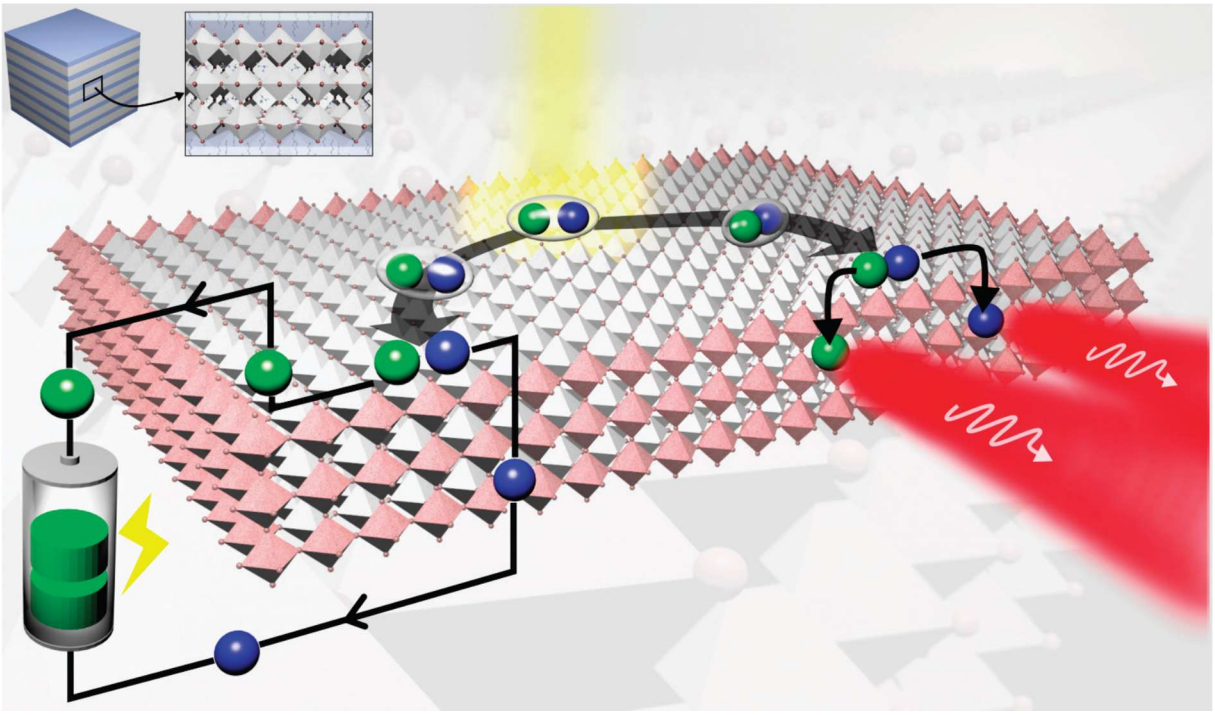
ستتطلب التقانة العملية مزيدًا من الجهود الهندسية لتصنيع هذه الخلايا على نطاق واسع ولاستقرارها على المدى الطويل. وهناك تساؤلاتٌ علمية ما زالت تحتاج إلى إجابة. فضلاً على ذلك، فإنّ الباحثين يعتقدون أن بإمكانهم تجنب السُميَّة البيئية للرصاص الموجود في المواد، وهُم قد لا يستطيعون تفادي وعي الجمهور لخطر وجودها.

ومع هذا فقد أعادت إمكانات البروفسكيتات إلى الواجهة تنظيم المجال.

**نُشِر هذا البحث في مجلة Communications of the ACM، المجلد 60، العدد 12، كانون الأول (ديسمبر) 2017، الصفحات 11 – 13.*

[1] التقانة الفوتوفولطائية (photovoltaic): هي تقانة تحويل الطاقة الشمسية إلى طاقةٍ كهربائية مباشرةً بواسطة الخلايا الشمسية. (المراجع)

[2] البروفسكيت: اسم يطلق على المواد التي لها البنية البلورية لفلز تيتانات الكالسيوم $CaTiO_3$، والصيغة الكيميائية العامة لهذه المواد ABX_3 حيث X أنيون الأكسجين أو أنيون الهالوجين وكلٌّ من A وB كاتيون. وقد اشتق الاسم من اسم مكتشفه فون بيروفسكي Von Perovski. (المترجم)



بِنية المادة وهي تبين تخطيطياً تكدُّس طبقات البروفسكيت الثنائية البعد ذات الثخانة النانومترية، والطبقات العضوية الفاصلة بينها.

انعطافٌ جديد

البروفسكيتات فصيلةٌ من المواد معروفةٌ جيداً، تتكون بلوراتها من شبيكة من ثمانيات وجوه، يشتمل كلّ منها على ستة أيونات سالبة موزعة حول أيون موجب، إضافةً إلى أيون موجب ثانٍ يتوضَّع في الفجوة بين ثمانيات الوجوه. يمكن أن تدور ثمانيات الوجوه بسهولة مع الإبقاءعلى تماسها بعضها مع بعض عند رؤوسها، وهذا ما يكسب البروفسكيتات خصائص غير اعتيادية، لطالما جرى استثمارها في تطبيقات مُتخصِّصة. فعلى سبيل المثال، تعمل نيوبات الليثيوم (Lithium niobate) محوالًا كهرضغطيًا (piezoelectric transducer) بين الإشارات الكهربائية والميكانيكية، كما تعمل كمعدِّلاتٍ ضوئية (optical modulators) عالية السرعة في مجال الاتصالات. يتبع ترتيب الأيونات في المواد الجديدة لهذه القواعد، بالرغم من أن مكوِّناتها مختلفة.

لقد جاءت أولى التلميحات بإمكانات الخلايا الشمسية المصنوعة من أنصاف النواقل البروفسكيتية من اليابان في عام 2009، إلاّ أن الباحثين انتبهوا فعليًا لذلك في عام 2012، عند نشر تقريرٍ عن خلايا شمسية بلغ مردودها قرابة 8%. في الحالة العامة، يكون الأيون الموجب الثاني في هذه البروفسكيتات الهجينة، المكوَّنة من مادةٍ عضوية وهاليد الرصاص اللاعضوي، جزيئًا عضويًا كبيرًا (يحتوي على الكربون)، في حين يتألف ثماني الوجوه من الرصاص (أو من القصدير أحيانًا) المحاط بأيونات الهالوجين (من الفصيلة VIIمن الجدول الدوري، وعادةً ما يكون عنصر اليود).

"ثمة أسباب، جعلت الناس لا يكترثون للأمر" كما يقول ميخائيل ماغهي (Michael McGehee)، وهو أستاذ علوم وهندسة المواد في جامعة ستانفورد بولاية كاليفورنيا. فلهذه المواد سمةٌ لاعضوية إضافةً إلى وجود أيونٍ عضوي فيها". إن أفضل المواد أداءً في الخلايا الشمسية هي السليكون حاليًا، وهي تصنع من شرائح أحادية البلورة كبيرة، مع أن هناك مواد لاعضوية أخرى تُستعمل أيضًا. إن أنصاف النواقل العضوية، المشابهة لتلك التي تستعمل في بعض شاشات الإظهار من نوع الديودات المُصدِرة للضوء (light emitting diodes, LED)، تصنَّع منها خلايا شمسية جيدة أيضًا، ويمكن توضيعها في درجات حرارة قريبة من درجة حرارة الغرفة. لكن البروفسكيتات الهجينة "مختلفة جدًا" عن كلا هذين النوعين من الخلايا الشمسية، كما قال ماغهي.

من الأهمية بمكانٍ ملاحظة أن البروفسكيتات الهجينة تضم صنفًا كاملًا من المواد، تبعًا للمركبات الداخلة في هيكل البلورة. ويستطيع الباحثون توليفَ الخصائص الضوئية والكهربائية، إضافةً إلى التآثرات الكيميائية التي تُشكِّل النبيطة كلها، وذلك بضم معادن وهالوجينات ومركبات عضوية مختلفة. يقول ماغهي "هناك عدة آلاف من المواد" وإذا أخذنا بالحسبان مواد الإلكترودات والسطوح البينية اللازمة لتصنيع نبيطة كاملة " فإنّ عدد التوفيقات (combinations) يغدو بالملايين."

حصاد الفوتونات

يتطلب استخراج الطاقة الكهربائية من ضوء الشمس بِنىً هندسيةً معقّدة. تُضَمُّ طبقاتٌ مختلفة لتوجيه ضوء الشمس نحو الطبقة نصف الناقلة في البداية، ولتمرير الطاقة الكهربائية المتولدة بعد ذلك إلى العالم الخارجي، وللحيلولة دون تفاعل الماء والهواء مع المواد المستعملة. إلَّا أنّ الدور المركزي ينطوي على خاصةٍ لنصف الناقل تُسمى فُرْجة الطاقة (bandgap)، أي مجال الطاقة الفاصل بين حالات الطاقة الدنيا المملوءة بالإلكترونات في البداية وحالات الطاقة الأعلى الخالية من الإلكترونات. يُمكن لفوتون، أو لِلْكمِّ الضوئي (quantum of light)، الذي طاقته أكبر من فرجة الطاقة أن يُحرِّر إلكترونًا شحنته سالبة إلى "عصابة الطاقة" الفارغة، مُخلِّفًا وراءه "ثقبًا" (hole) شحنته موجبة.

إذا شق الإلكترون والثقب طريقهما إلى مسريين متعاكسين متصلين بالمادة، فإنهما يزودان العالم الخارجي بتلك الطاقة على شكل تيارٍ كهربائي. تكون الفلطية التي تُزَوِّد التيار محدودةً بفرجة الطاقة (مقدرةً بالإلكترون فولط). وإذا كانت طاقة الفوتون أكبر من تلك الطاقة، فإن الفائض يضيع على شكل حرارة، في حين تعبر الفوتونات التي لا تكفي طاقتها لتجاوز فرجة الطاقة المادة دون أن تمتصها. ولمّا كان ضوء الشمس يشتمل على فوتونات ذات مجالٍ واسع من الطاقة، فإن خرج الطاقة الكهربائية لخلية شمسية واحدة لا يمكنه نظريًا أن يتجاوز 33% من الطاقة الضوئية الواردة عليها.

مع ذلك، فإن المردود الفعلي أخفض دائمًا من هذا الحدّ النظري. إنّ أحد أسباب ذلك، هو أنه يمكن للإلكترون أن يعيد مَلْءَ حالةٍ طاقية شاغرة "فيعود للاتحاد" مع ثقبٍ موضعيًا، بدلًا من أن يعبر إلى الدارة الخارجية. تحدث هذه العملية بصورة أسرع عند وجود حالات إلكترونية في فرجة الطاقة، وهي مجال طاقة ينبغي أن يكون شاغرًا في الحالية المثالية. تكمن الميزة الرئيسية للمواد البروفسكيتية الهجينة في احتوائها على عددٍ قليل من العيوب الفعَّالة كهربائيًا. إنّ العمر الطويل للإلكترونات قبل العودة للاتحاد الناتج عن ذلك، إضافةً إلى الحركة السريعة للإلكترونات والثقوب، يُسهِّلان الاقتراب من المردود المثالي.

يظن "كاهن" (Cahen) أن ندرة العيوب في البروفسكيتات يعود إلى بِنيتها البلورية اللينة والمرنة، والتي يُنظر إليها على أنها "غير منتظمة ديناميكيًا،" (dynamically disordered) ولربما تتمتع "بالشفاء الذاتي (self-healing)." بالمقابل،

يقول ماغهي إن "المادة مملوءة بالعيوب"، لكن العيوب ليست مشكلةً بسبب البِنية الكيميائية للمادة. فقد لاحظ أنه في العديد من أنصاف النواقل، يمكن للمرء أن ينظر إلى فرجة الطاقة على أنها تعكس الفاصل الطاقي بين تراكيب الحالات "الرابطة" bonding والحالات "المضادة للربط" (antibonding) في الذرات المتجاورة، لذا تُولِّد الروابط المكسورة عند عيبٍ ما حالاتٍ تقع طاقتها في منتصف فرجة الطاقة، حيث يكون ضررها أعظميًا. بالمقابل، يقول ماغهي، يكون لفرجة الطاقة في البروفسكيتات منشأٌ مختلفٌ، ولا تُعزِّز حالات العيوب العودةَ للاتحاد.

مهما يكن التفسير النظري للأمر، فقد دفع المُجرِّبون مردود خلية شمسية بروفسكيتية وحيدة إلى قيمة أعلى من 22% بالفعل، وهذا ما يجب مقارنته بالقيمة 27% وهو المردود الذي حققته أفضل الخلايا السليكونية البلورية بعد عقودٍ من العمل.

تصنيعٌ على البارد

إن للبروفسكيتات الهجينة مزايا مفيدة أخرى تُضاف إلى مردودها العالي: فمكوِّناتها وفيرة، ويمكن تشكيل أغشية رقيقة منها في درجة حرارة الغرفة بعمليات تعتمد على السوائل أو الأبخرة. هذه الخصائص تُقلِّل تكلفة تصنيعها مقارنةً بتنمية بلورات السليكون في درجات حرارة عالية جداً. يُحذِّر هنري سنيث (Henry Snaith)، وهو فيزيائي من جامعة أكسفورد، من أن الجودة والانتظام لهما على الأقل نفس أهمية تكلفة التجهيزات اللازمة لإنتاجها على نطاق واسع.

تسمح المعالجة في درجة حرارة منخفضة أيضًا بترسيب أغشيةٍ رقيقة فوتوفولطائية على سطوحٍ لا تتحمل درجات حرارة عالية. على سبيل المثال، تشمل التطبيقات الناشئة المعروفة باسم الفوتوفولطائيات المُدمجة في الأبنية (building-integrated photovoltaics, BIPV) خلايا شمسية مُرسَّبة مباشرةً على بلاط الأسقف (roof shingles) أو على زجاج نوافذ الأبنية البرجية المكتبية، حيث تُولِّد الطاقةَ وتدعُ شيئًا من الضوء يمر خلالها.

لقد ألهمت هذه التطبيقات سنيث (Snaith) إنشاءَ شركةٍ أطلق عليها اسم فوتوفولطائيات أكسفورد (Oxford photovoltaics) في عام 2010، لتطوير فوتوفولطائيات عضوية مُحَسَّسة بالصباغ (dye-sensitized)، تستعمل جزيئاتٍ لامتصاص الضوء (الصباغ)، وجزيئاتٍ من مادةٍ أخرى لنقل التيار الكهربائي (يمكن ترسيبها أيضاً في درجات حرارةٍ منخفضة). ولكن، بعد أن أحرز فريقه في الجامعة وآخرون نتائج واعدة، "حوَّلت شركته سريعًا كلَ نشاطاتها تجاه البروفسكيتات." كما تحوَّلت الشركة من السوق "الصغير" للفوتوفولطائيات المُدمجة في الأبنية إلى تطبيقات اللوحات الشمسية المتداولة، كما يقول سنيث. "لقد أدركنا مبكِّرًا أنه يمكن للبروفسكيتات زيادةَ مردود السليكون في الخلايا الشمسية الترادفية (tandem)"[3]، وذلك بالإفادة من العمليات المنخفضة درجة الحرارة لتصنيع خلية بروفسكيتية فوق خلية سليكونية بلورية مصنوعة سابقًا.

يمكن للخلايا الشمسية الترادفية استخراج طاقةٍ أكبر من طيف الإشعاع الشمسي، وبالإمكان أن يتجاوز مردودها المردودَ الحدّي لأية خلية شمسية وحيدة. باستعمال مادة ذات فرجة طاقية أكبر، تتمكن الخلية البروفسكيتية من الإفادة أكثر من جزء الطاقة العالية للطيف الشمسي، في حين تنفذ الفوتونات ذات الطاقة الأخفض إلى السليكون لتوليد فلطية أكثر مواءمةً لعملها. تشير الدراسات المنشورة عن الخلايا الشمسية الترادفية البروفسكيتية/السليكونية، إلى أن مردودها بلغ 23.6% (ما

[3] الخلايا الشمسية الترادفية: تعتمد إحدى طرائق زيادة مردود الخلايا الشمسية على تقسيم الطيف الشمسي واستعمال خلية شمسية مثلى لكل قسم من الطيف، وهذا ما يُسمى بالخلايا الشمسية الترادفية. يمكن أن تكون هذه الخلايا خلايا منفردة أو مربوطة على التسلسل. إنّ النوع الأول هو الأكثر شيوعاً، حيث تُرسّب كل الخلايا على شكل طبقات متتالية على ركازة واحدة. (المراجع)

يقارب 26% عند وصل الخلايا بشكلٍ منفصل). ثمة منهجية بديلة، تقوم على ضمِّ مادتين بروفسكيتين لكلٍ منهما فرجة طاقة مختلفة عن الأخرى، وقد بلغ مردودها 20.3% عند وصلهما بشكلٍ منفصل.

توسيع الإنتاج (scaling up)

بالرغم من التقدم السريع للبحوث في مجال الخلايا الشمسية المعتمدة على المواد البروفسكيتية، فإن هناك حاجة لمزيدٍ من العمل للوصول إلى تقانة يمكن تسويقها تجاريًا. إذ تتطلب اللوحات[4] العملية أغشية كبيرة ومنتظمة وذات جودة عالية، إضافةً إلى داراتٍ مساعدة تمنع إعاقة الخلايا لعمل كامل اللوحة. لتحقيق هذا الهدف، يقول سنيث "إن شركة فوتوفولطائيات أكسفورد استحوذت على منشأة في براندنبرغ (Brandenburg) في ألمانيا، وتسعى لبناء خط إنتاج تجريبي باستعمال شرائح سليكونية قطرها ست بوصات، في نهاية عام 2018. وهناك شركاتٌ أخرى ليست بعيدةً عن تحقيق ذلك.

لمّا كانت البروفسكيتات تتفاعل مع الماء والهواء، فإن الخلايا الشمسية التجارية تحتاج على الأغلب إلى مستَوَيات متعددةٍ من الحماية أو التغليف. يتطلب الزبون كذلك ضمانًا إضافيًا بأن اللوحات التي تستعمل هذه المواد الجديدة ستكون مستقرةً على امتداد سنوات استعمالها المديدة.

ثُمّ إن الباحثين ما زالوا يتحرَّون عن المكوِّنات التي تعطي أفضل نصف ناقل (فضلاً على مختلف المواد المستعملة في المساري والسطوح البينية والتغليف). يقول ماغهي "إذا كان لي أن أتنبأ، فسأقول إننا لم نعثر بعد على هذه المادة." إلّا أنّ سنيث أشار إلى أن شركته استقرت على تركيبةٍ مُعيَّنة في تصنيعها الابتدائي، لكنه يتوقع أنّ تتغير هذه التركيبة بمرور الزمن.

ثمة مُكوِّن تبين أنه من الصعب استبداله وهو الرصاص، الذي يتمتع بسمعةٍ سيئةٍ مُستحقّة، بسبب الآثار الصحية الخبيثة الناجمة عن استعماله السابق في البنزين والطلاء. يقول (كاهن): "لم نستطع حتى الآن إيجاد أبدال يقارب أداؤها، ولو من بعيد، أداءَ هذه المواد من دون الرصاص". (حقَّق الباحثون بعض النجاح باستعمال القصدير بدلاً من الرصاص، إلّا أنّ هذه المواد لا تزال تعاني من مشكلة الاستقرارية).

لمّا كانت ثخانة طبقة البروفسكيت اللازمة أقل من المِكرون، مقارنةً بمئات المكرونات من السليكون، فإن الكمية الإجمالية للرصاص في اللوحة ضئيلةٌ جدًّا. إضافةً إلى ذلك، فإن التغليف اللازم للاستقرارية سيُبْقي الرصاص في مكانه. يقول سنيث "مع أنّ حساسيتنا تجاه الرصاص شديدةٌ، فإني أعتقد أن احتمال أي خطرٍ بيئي حقيقي من وجوده ضئيلٌ جداً، شريطة أن تُركَّب اللوحات على نحوٍ معقول، وأن يُعاد تدويرها بعد انتهاء عمرها." ومع ذلك، فهو يعترف أن تَقَبُّلَ الجمهور للأمر قد يبقى مشكلةً بحاجة إلى حل.

يقول ماغهي " لا يعني هذا أنك لن ترى هذه الخلايا في الأجهزة المحمولة." وهو يتكهَّن أن الحصول على مردود 30% للَّوحات المعتمدة على تصاميم خلايا ترادفية مُجدية التكلفة أمرٌ ممكنٌ جداً، في حين أن مردود اللوحات السليكونية الأعظم سيبقى محصورًا بين 22 و23%.

[4] بما أن الفلطية والاستطاعة التي تولِّدهما خلية شمسية واحدة صغيرتان نسبياً، فإنه يجري عادةً ضمُّ مجموعةٍ من الخلايا على التسلسل و/أو التفرع في لوحة لتوليد فلطية واستطاعة مقبولتين؛ ويجب أن تكون الخلية الواحدة كبيرة السطح لإنقاص تكلفة التصنيع. (المراجع)

قراءة إضافية

- *Shockly, W., and Queisser, H.J.*, Detailed Balance Limit of Efficiency of p-n Junction Solar Cells, *J. Appl. Phys.* 32, 510 (1961). http://dx.doi.org/10.1063/1.1736034
- *Bush, K.A., et al.*, 23.6%-efficient monolithic perovskite/silicon tandem solar cells with improved stability, *Nat. Energy* 2, 17009 (2017). http://dx.doi.org/10.1038/nenergy.2017.9
- Perovskite Cells for Tandem Applications, École Polytechnique Fédérale de Lausanne, http://pvlab.epfl.ch/page-124775-en.html

الكاتب

دون مونور (Don Monore) كاتب عن العلوم والتقانة، مقره في بوسطن مساشوستس بالولايات المتحدة الأمريكية.

قائمة المصطلحات

LIST OF TERMS

إعداد: د. نزار الحافظ

ASIC (application-specific integrated circuit)	دارة متكاملة محددة بحسب التطبيق
asynchronous handoff	تناقلُ لامتزامن
audio and video conferencing	ائتمارٌ مسموعٌ ومرئيٌّ
auditability	تدقيقية
autocomplete menus	قوائم الإكمال التلقائي
automata processor	معالج الأتومات
beta-tester	المختبِر بيتا
block language	لغة كتل
blocks-oriented interface	واجهة كتلية التوجُّه
chunking code	تقطيع الرماز
closed captioning	عروض نَصِّيَّة مغلقة
CMOS (Complementary metal–oxide–semiconductor)	معدن–أكسيد–نصف ناقل متتامين
cognitive load	حِمل معرفي
compartmentalization of software	تقسيم البرمجيات إلى أجزاء مستقلة
computer-augmented aid	مساعَدة مزيدة بالحاسوب
concatenative speech synthesis	تركيب الكلام بالضم
cross links	روابط تقاطعية
dark net	شبكة مظلمة
drag-and-drop	سَحْب وإفلات
DSP (digital signal processor)	معالج الإشارة الرقمية
DVFS (dynamic voltage and frequency scaling)	تدرج ديناميكي للفلطية والتردد
epiphenomenon	ظاهرةٌ ملحَقة

exascale computing	الحوسبة برتبة إكسا عملية بالثانية
fat cores	نوى سمينة
FPGA (field programmable gate array)	صفيفة بوابات قابلة للبرمجة حقلياً
GPU (graphic processing unit)	وحدة معالجة بيانية
hand-off (adj, noun)	تناقليّ، تناقُل
heterogeneous computing	حوسبة لامتجانسة
hyperthreading capacity	قدرة النيسبة الفائقة
infotainment	إعلام وترفيه
in-line autocomplete	إكمال تلقائي مضمَّن
interoperability	تشغيلية بينية
intonation	نبرة الكلام
keywords	كلمات رئيسية
LiDAR (Light Detection And Ranging)	كشف الضوء وتقدير المدى
linear output	خَرْج خَطِّيّ
liveness	رشاقة
multicore systems	نظم متعددة النوى
multivendor	مورِّدون متعددون
neuromorphic chips	رقاقات عصبونية شبكية
nonvolatile memory	ذاكرة لامتلاشية
omniscient debugging	تفليةٌ كلّيةُ العِلم
online	موصول بالخط
OpenCL (open computing language)	لغة الحوسبة المفتوحة
out-of-order execution	تنفيذ خارج الترتيب
parallel simultaneous editing	تحريرٌ متواقتٌ مُوازٍ
parsing	تحليل نحْويّ
Phoneme(s)	صَوتِيم(ات)
photovoltaics	فوتوفولطائيات
pointing & clicking	تأشير ونقر
prosody	تنغيم

readability	مَقروئية
recall	تذكُّر
recognition	تمييز
research prototypes	نماذج بحثية أوَّلية
revision history	تأريخ المراجعة
runtime	وقت التنفيذ
sci-fi	أمانة علمية
selection locking	إقفال انتقائي
self-healing	شفاء ذاتي
semantic errors	أخطاء دلالية
serial handoff	تناقُل متسلسل
simultaneous editing	تحرير متواقت
simultaneous writing	كتابة متواقتة
speculative execution	تنفيذ تحزّري
speech synthesis	تركيب الكلام
superscalar width	عرض السُلَّمية الفائقة
syllable	مقطع صوتي
syntax	التركيب النحْوي
telescope	مِقْراب (ج: مَقاريب)
TFET (tunneling field effect transistor)	ترانزستور مفعول الحقل النفقي
timeline visualization	عرض مرئي للخط الزمني
user friendliness	سهولة الاستعمال
vowel	صائت

مطبوعات الجمعية العلمية السورية للمعلوماتية			
الكتب التخصصية	السعر (ل. س)		
	أفراد	أعضاء جمعية وطلاب	مؤسسات
معجم مصطلحات المعلوماتية	2000	1600	4000
أسس لغات البرمجة	1600	1200	2000
هندسة البرمجيات – المجلد الأول	1800	1200	2400
هندسة البرمجيات– المجلد الثاني	1400	800	2000
الذكاء الصنعي	1500	1000	2000
مفاهيم نظام التشغيل– الجزء الأول (تجليد عادي)	1600	1000	-
مفاهيم نظام التشغيل– الجزء الأول (تجليد فنّي)	1900	1300	2200
مفاهيم نظام التشغيل– الجزء الثاني	1600	1100	1900
التعمية التطبيقية (Applied Cryptology)	2400	1600	3000
المدخل إلى Mathematica 5.0 (تجليد عادي)	600	400	-
المدخل إلى Mathematica 5.0 (تجليد فني)	800	600	1200
اتصالات المعطيات والحواسيب – الجزء الأول	1600	1100	1850
اتصالات المعطيات والحواسيب – الجزء الثاني	1400	1000	1650
مسرد مصطلحات المعلوماتية إنكليزي – عربي	300	200	500
مجلة الثقافة المعلوماتية	250	200	600
مدخل إلى الخوارزميات – الجزء الأول	1800	1200	2400
مدخل إلى الخوارزميات – الجزء الثاني	1800	1200	2400
أسس نظم قواعد المعطيات – الجزء الأول	1800	1800	1800
البرمجة المتوازية – تقنياتها وتطبيقاتها باستعمال محطات عمل شبكية وحواسيب متوازية			